Universidad Carlos III de Madrid
Escuela Politécnica Superior

GRADO EN INGENIERÍA TELEMÁTICA

TRABAJO FIN DE GRADO

IMPLEMENTACIÓN DE UN MÓDULO PARA LA DETECCIÓN DE SENTIMIENTOS EN LA PLATAFORMA KHAN ACADEMY

Autor:
David Arellano Martín-Caro
Tutor:
Pedro José Muñoz Merino
Co-tutor:
Derick Leony

Leganés, Febrero 2014

AGRADECIMIENTOS:

Lo primero de todo agradecer a Pedro la oportunidad que me ha dado de trabajar en una plataforma de uso educativo, a Derick por haber dedicado tiempo a mi proyecto, y a Jose por haber estado siempre ahí cuando he tenido alguna duda.

Este período de mi vida en la universidad ha sido largo y a veces difícil, por eso agradecer a todos los conocidos que me deja esta etapa, tanto amigos como compañeros de prácticas, el hacerme el trabajo un poquito más fácil en los momentos malos.

También agradecer a los amigos que tengo desde que soy pequeño el haberme animado a estudiar, aunque no haya estado con ellos todos el tiempo que debiera por habernos separado al llegar esta etapa de la vida, en la que cada uno elige estudiar una cosa y nos vimos obligados a separarnos un poco.

Y por último, y no menos importante, agradecer a toda mi familia el apoyo mostrado durante todo este período, pues sin ellos no sé si hubiera podido llegar hasta aquí.

Gracias a todos.

RESUMEN:

Cada vez se utilizan más las plataformas educativas para que un profesor pueda hacer el seguimiento de un grupo de alumnos. Estas plataformas guardan una gran cantidad de datos de bajo nivel que contienen mucha información sobre la actividad de los alumnos, pero que son difícilmente interpretables. En este trabajo se ha utilizado una versión off-line de la plataforma Khan Academy para poder extraer estos datos de bajo nivel y pasarlos a información de alto nivel. En concreto, estos datos de alto nivel han servido para inferir, mediante algoritmos, los sentimientos (frustración, confusión, alegría y aburrimiento) de los alumnos a partir de los datos de bajo nivel. También se han desarrollado visualizaciones para ver el nivel de cada sentimiento de cada alumno y de la media de la clase. Las visualizaciones muestran el nivel de cada uno de los sentimientos (medidos de cero a uno) de cada alumno registrado en la plataforma, respecto a unas fechas determinadas que se irán generando automáticamente de forma periódica para que las medidas siempre estén actualizadas. El nivel de cada sentimiento de un alumno en una visualización vendrá acompañado del nivel de ese mismo sentimiento de la media de la clase, y así podrá ser comparado. A través de estas visualizaciones el profesor podrá hacer un seguimiento de los alumnos, permitiéndole saber qué temas resultan más difíciles o cuáles resultan más amenos, etc., y así ayudar al proceso de aprendizaje de los alumnos.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

INTRODUCCIÓN

En este capítulo se van a mostrar las motivaciones para la realización del trabajo, los objetivos propuestos y la estructura de la memoria.

1.1. Motivación

Durante los últimos años las plataformas educativas se utilizan de una forma más frecuente. Una de las razones con más peso para la introducción de esta tecnología es la de tener una analítica de aprendizaje (learning analytics) potente, y así llevar a cabo un seguimiento de los alumnos.

El paso de datos de bajo nivel a datos de alto nivel para obtener información que sirva de ayuda en el proceso de aprendizaje de los alumnos, es uno de los grandes desafíos de learning analytics.

La plataforma Khan Academy ha sido elegida por la Universidad Carlos III de Madrid para algunos de sus cursos 0. Esta plataforma proporciona muchos datos de bajo nivel, los cuales va guardando a medida que los alumnos hacen ejercicios, ven vídeos, etc. El módulo ALAS-KA [1] es una extensión de la Khan Academy en la cual se apoya para pasar estos datos de bajo nivel a datos de alto nivel.

El trabajar en un proyecto donde poder ayudar a entender mejor los sentimientos de los alumnos a través de sus interacciones con la plataforma, y así poder mejorar el proceso de aprendizaje, además de ayudar a los profesores con sus métodos de enseñanza, han sido la razón de más peso a la hora de elegir este proyecto.

1.2. Objetivos

El objetivo principal de este Trabajo fin de Grado es la implementación de un módulo para la detección de sentimientos de la plataforma Khan Academy. Este módulo tendrán en concreto cuatro sentimientos a representar, que son, frustración, confusión, alegría y aburrimiento.

Estos sentimientos son los datos de alto nivel conseguidos a través del procesado de datos de bajo nivel [2] que la plataforma guarda cuando los alumnos interactúan. El

objetivo es conocer cómo se sienten los alumnos tras haber realizado una serie de ejercicios, donde, por poner un ejemplo, si el último ejercicio realizado por un alumno ha tenido una respuesta errónea, es normal que se sienta más frustrado que alegre.

Mediante visualizaciones se mostrará el nivel que tiene cada alumno en cada uno de los cuatro sentimientos, medidos de cero a uno. Mediante estas visualizaciones se puede saber qué sentimiento predomina con respecto a los demás en un momento dado. Estas medidas de los sentimientos son información de alto nivel.

Todo esto se va a desarrollar de forma independiente a la plataforma Khan Academy. Se va a trabajar en una versión off-line de la plataforma Khan Academy en la cual habrá que crear usuarios y generar datos para poder tener situaciones reales. Una vez se hayan recogido estos datos, se procesen para convertirlos en información de alto nivel y se creen las visualizaciones correspondientes, habrá una fase de integración en el módulo "Add-on for the Learning Analytics Support in the Khan Academy platform" (ALAS-KA).

1.3. Estructura de la memoria

La memoria se encuentra dividida en 6 capítulos descritos a continuación:

Capítulo 1. Introducción: En este capítulo se hace una breve introducción, se muestran las motivaciones que llevaron a realizar este proyecto y se hace un comentario sobre los objetivos del proyecto.

Capítulo 2. Estado del arte: Este capítulo comienza con una introducción a la plataforma educativa Khan Academy, seguido de un comentario sobre learning analytics ,como está presente en la Khan Academy y el módulo ALAS-KA. Se hace un repaso a algunos estudios sobre el reconocimiento de emociones, se resumen las tecnologías utilizadas en el proyecto y por último se explica la versión off-line de la plataforma Khan Academy que es donde se ha trabajado.

Capítulo 3. Modelos de detección de sentimientos en la plataforma Khan Academy: En este capítulo se mostrarán los diagramas de flujo seguidos para la detección de sentimientos y se explicará cada uno de los pasos.

Capítulo 4. Implementación de detector de sentimientos para la plataforma Khan Academy: En este capítulo se muestra un diagrama de clases donde las clases están agrupadas por niveles según su función, se explicarán las clases creadas para el desarrollo del proyecto y se explicarán las visualizaciones conseguidas en la versión off-line de la plataforma previas a la integración en el módulo ALAS-KA.

Capítulo 5. Integración, evaluación y validación: Este capítulo habla del proceso de integración del módulo de detección de sentimientos en el módulo ALAS-KA y muestra visualizaciones sobre la evaluación y validación de las pruebas.

Capítulo 6. Conclusiones y trabajo futuro: Este capítulo cierra la memoria comentando las conclusiones extraídas del trabajo realizado y muestra futuros trabajos que se pueden acometer a partir de este.

CAPÍTULO 2

ESTADO EL ARTE

En este capítulo se va a hacer un repaso a la plataforma Khan Academy, la funcionalidad learning analytics, un reconocimiento del modelo para la detección de sentimientos y se van a explicar las tecnologías utilizadas.

2.1. La plataforma Khan Academy

Actualmente, el término MOOC (Massive Online Open Course) ha tomado más fuerza. La aparición de muchos de ellos por universidades de todo el mundo ha servido para dar un salto de calidad en el sector del e-learning. La particularidad de estos cursos, por la que ha tenido gran aceptación y cada vez están más extendidos, ha sido una característica que los diferencia de lo ya habitual: la posibilidad de juntar al mismo tiempo a mucha gente, bajo un mismo calendario, siendo accesible para miles de personas al mismo tiempo. Algunas de las plataformas denominadas MOOC son Coursera, Udacity, edX o Khan Academy.

La Khan Academy[3] es una organización sin ánimo de lucro que pretende poner a disposición de cualquier persona una educación de manera totalmente gratuita, además de proporcionar visualizaciones de learning analytics.

Se va a realizar un repaso por la plataforma Khan Academy, donde cada figura vendrá acompañada de una numeración asociada a un área facilitando la explicación.

La figura 1 muestra la página de inicio de la Khan Academy en su versión en inglés:

1- Botón de logeo. Los usuarios que ya se han registrado pueden autenticarse directamente.
2- Posibilidad de registro directo si se es usuario de Facebook o si se tiene una cuenta de Google y uso de estas dos vías para el aviso de notificaciones. También registro como profesor para llevar el seguimiento de un aula, o como padre para llevar el seguimiento de tus hijos.
3- Menú donde se pueden ver todos los vídeos y ejercicios que hay e incluso realizarlos, pero sin que se guarden los datos que se van generando. También se puede ver un manual para tutores y entrevistas a los fundadores.

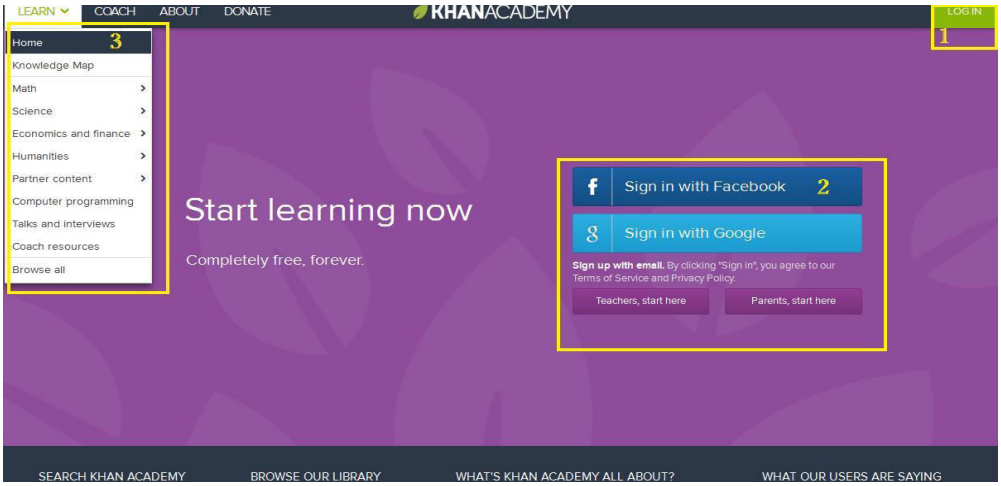

Figura1. *Página de inicio de la Khan Academy [2]*

En la figura 2 se ve el escritorio personal del usuario que se ha registrado:

1- Menú personal donde cambiar el perfil, ver los badges ganados, visualizaciones con tus estadísticas, etc.
2- Nombre del usuario registrado.
3- Notificaciones pendientes de ver.
4- Energy points ganados por el usuario (son el tipo de puntos que utiliza Khan Academy).
5- Muestra los badges ganados de cada tipo, y es un acceso directo al menú de badges

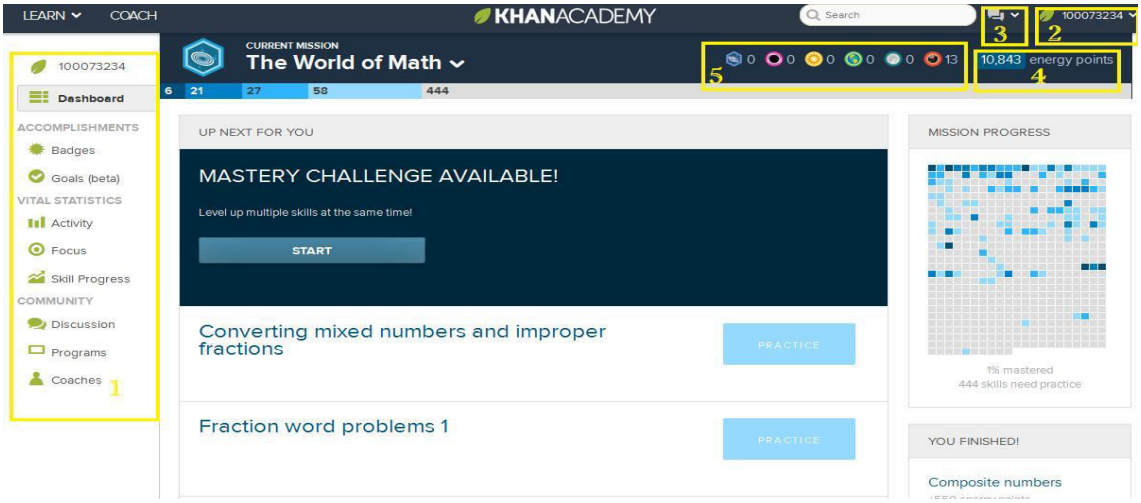

Figura2. *Escritorio personal [2]*

En la figura 3 se muestra el knowledge map, donde podemos ver todos los ejercicios que están disponibles. Los ejercicios están relacionados entre sí de manera que haga más fácil el aprendizaje:

1- Menú donde los ejercicios están relacionados por un sistema recomendador.

2- Buscador de ejercicios.

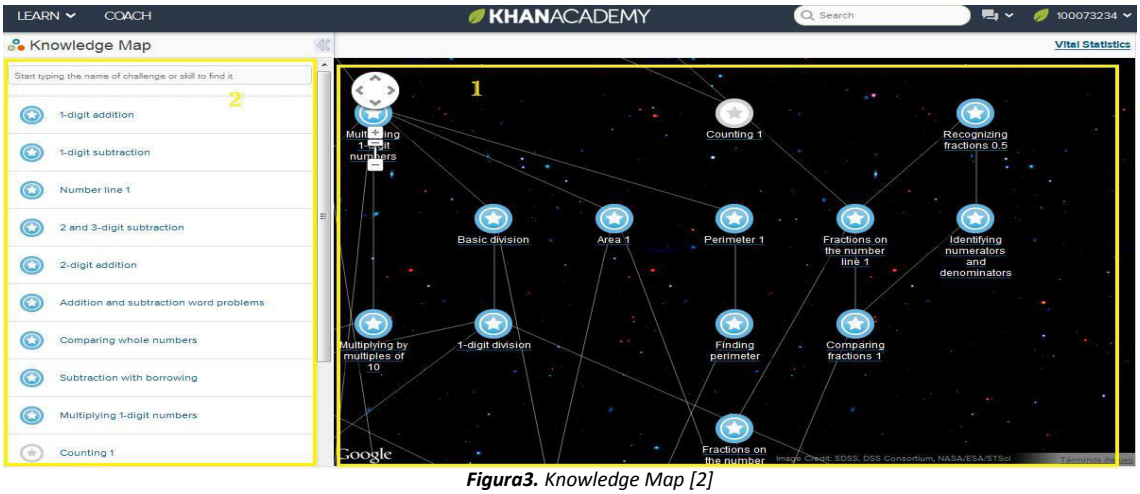

Figura3. *Knowledge Map [2]*

En la figura 4 se muestra la interfaz para la realización de un ejercicio. Si un ejercicio es resuelto correctamente se mostrará otro enunciado diferente, si la solución dada es errónea se mantendrá el mismo enunciado hasta que se resuelva correctamente.

1- Nombre del ejercicio elegido.
2- Enunciado del ejercicio.
3- Campos para la dar la respuesta al ejercicio, junto con el botón de comprobar la respuesta.
4- Campo donde se puede pedir una pista para la resolución del ejercicio.
5- Ventana que muestra el vídeo relacionado con el ejercicio, lo que proporciona al usuario la posibilidad de ver el vídeo en caso de no saber resolver el ejercicio.
6- Contador de respuestas correctas consecutivas. Una vez llegues a 5, se obtiene el nivel de proficiency en ese ejercicio.

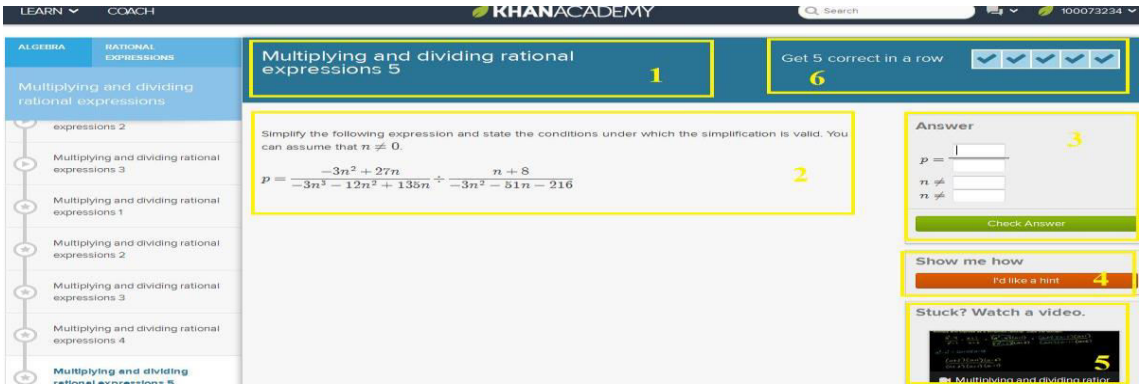

Figura4. *Interfaz para la resolución de un ejercicio [2]*

En la figura 5 se muestra la interfaz para el visionado de vídeos:

1- Nombre del vídeo que se está viendo.

2- Representa la cantidad de energy points conseguidos por la cantidad de video visionado.
3- Campo donde se pueden formular preguntas que pueden ser respondidas por otros usuarios.

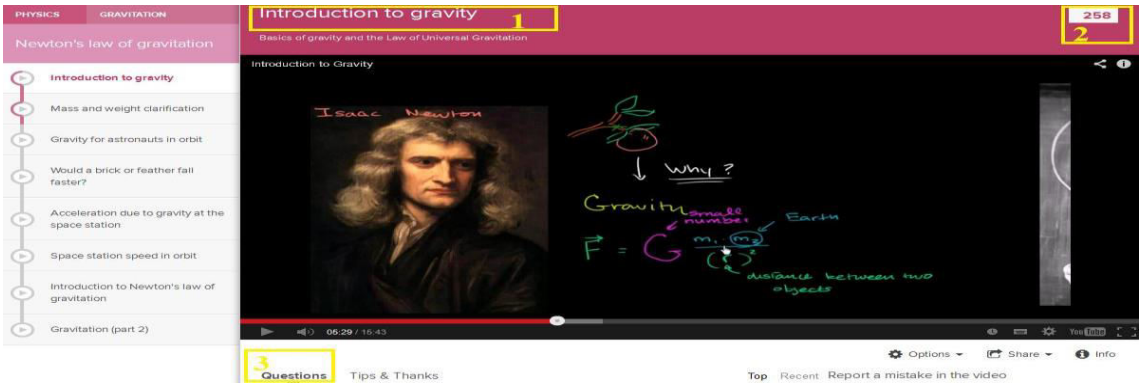

Figura5. *Interfaz para el visionado de vídeos [2]*

En la figura 6 se ve el menú de los badges de cada usuario:

1- Badges ganados por el usuario.
2- Posibles badges a ganar por el usuario, siendo los primeros los más posibles.
3- Cantidad de veces que se ha ganado un mismo badge.

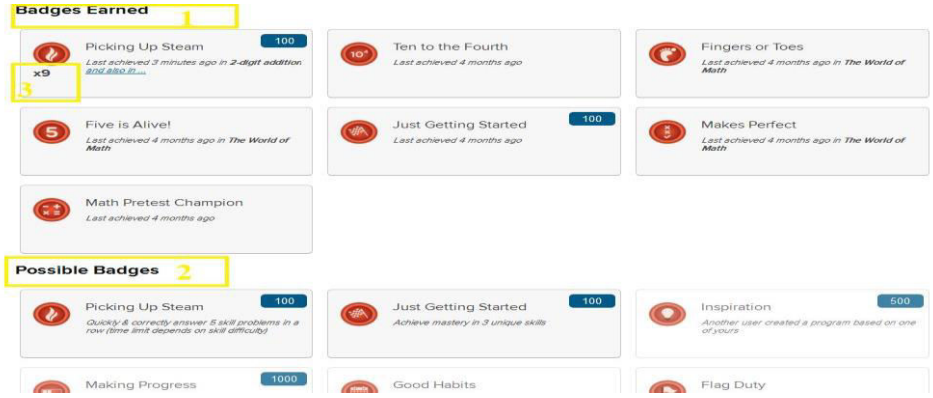

Figura6. *Menú de Badges [2]*

Otra de las posibilidades de la plataforma Khan Academy es la de ponerse Goals, es decir, el usuario se autoimpone unos objetivos a completar, siendo estos objetivo unas combinación de vídeos, ejercicios, o vídeos y ejercicios a la vez.

2.2. Learning Analytics

2.2.1. Introducción a learning analytics

La forma de imaginar el futuro de la educación cada vez se acerca más a las nuevas tecnologías de dispositivos (diseños flexibles de clase y visualizaciones

innovadoras). Pero, sin embargo, el factor más determinante para la configuración de la educación en el futuro es algo que no podemos tocar ni ver: Big Data y Analytics.

El término Big Data se usa para describir un nuevo contexto de abundancia. El Instituto Global McKinsey define Big Data como "conjunto de datos cuyo tamaño va más allá de la capacidad de las herramientas de software de bases de datos típicas para capturar, almacenar, gestionar y analizar" [4].

El Big Data se encuentra presente en muchos sectores, y cada uno intenta utilizarlos de la forma que más rendimiento genere a los intereses propios. Algunos ejemplos del uso de Big Data son:

- Publicidad que aparece en páginas webs y que está relacionada con anteriores búsquedas en Google.
- Recomendadores en redes sociales.
- Almacenamiento de datos en centros de estudios de meteorología.
- Sector del e-learning

Se puede definir como learning analytics a la aplicación de Big Data en el sector del e-learning. Una definición del término learning analytics en un sentido amplio puede ser esta [4]: *El aprendizaje es el análisis de la medición, recolección, análisis y reporte de datos sobre los estudiantes y sus contextos, a los efectos de la comprensión y el aprendizaje de optimización y los entornos en que se produce*.

En 54] se define el proceso de aprendizaje en cinco pasos: Capturar, Informar, Predecir, Actuar y Refinar. Como indica el tercer paso de esta definición, este modelo gira en torno a la fase de predicción, con lo cual no concuerda del todo con el trabajo desarrollado en este proyecto, pues no se aplican modelos predictivos.

Según [6], un proceso de análisis de aprendizaje se puede dividir en tres etapas:

1- La interacción de un usuario con la plataforma genera una gran cantidad de datos de bajo nivel. El almacenamiento de todos estos datos puede considerarse el primer paso.
2- El análisis y procesamiento de estos datos de bajo nivel para transformarlos en datos de alto nivel y poder obtener información de ellos, se considera el segundo paso. El objetivo de esto es mejorar la experiencia de aprendizaje tanto del instructor como del aprendiz.
3- Las intervenciones de un profesor a un alumno después de haber visto la información que proporciona el segundo paso, se considera el tercer paso. Por ejemplo, si de los datos de alto nivel, hemos visto que un alumno está

frustrado, el profesor podría intervenir con una explicación del tema que está frustrando al alumno.

En [7] se define el análisis de aprendizaje como un ciclo de cuatro pasos. El ciclo no termina, e incluso puede ser completado cuando la intervención no llega a los alumnos. Un ejemplo muy sencillo para ver como se completaría un ciclo, sería cuando un profesor revisa las calificaciones finales de un curso y enseña las notas finales a toda la clase.

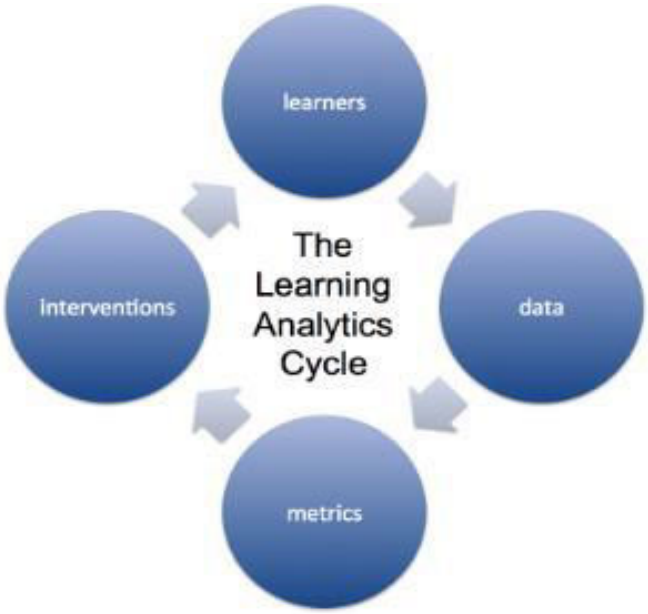

Figura7. *Ciclo learning analytics [6]*

En la figura 7 se muestran los cuatro pasos del ciclo de analítica de aprendizaje, en el cual los estudiantes (primer paso) generan datos (segundo paso), se procesan las métricas (tercer paso) y se cierra el ciclo con las intervenciones (cuarto paso). A continuación se pasa a explicar más en detalle cada uno de los pasos:

- Learners: Es el comienzo del ciclo. Los learners pueden ser estudiantes en un curso de la universidad o alumnos tomando partes de un MOOC.
- Data: Este paso se compone de la generación de datos por parte de los alumnos y de la captura de estos datos.
- Metrics: En este paso se procesan los datos para realizar un análisis. Esto incluye visualizaciones, tablas de control, comparaciones de resultados con puntos de referencia, etc. Este paso es el más importante en la mayoría de proyectos de analítica de aprendizaje y ha sido el centro de innovación en herramientas, métodos y metodologías.
- Interventions: Sin este paso el ciclo no se completa. Las intervenciones son indicadores que tienen que tener algún efecto sobre los estudiantes.

2.2.2. Learning analytics en la Khan Academy

La Khan Academy ofrece un gran sistema de analítica de aprendizaje. Algunas de las características que proporciona están relacionadas con el progreso de habilidades, el informe del ejercicio, o el informe de actividades de los estudiantes. Estas características están disponibles a través de visualizaciones, donde un usuario puede ver sus propias estadísticas, y donde un tutor o profesor puede ver las estadísticas de toda la clase o de un alumno por separado.

En la figura 8 se muestra el informe *Activity* que muestra cuál ha sido el tiempo dedicado a ejercicios y vídeos y los puntos obtenidos. Se tiene la opción de escoger el período de tiempo en el cual queremos ver este informe, pudiendo elegir el día actual, el anterior, el último mes, etc. Poniendo el cursor encima de cualquiera de las barras se amplía la información sobre los vídeos visionados, ejercicios realizados y badges ganados.

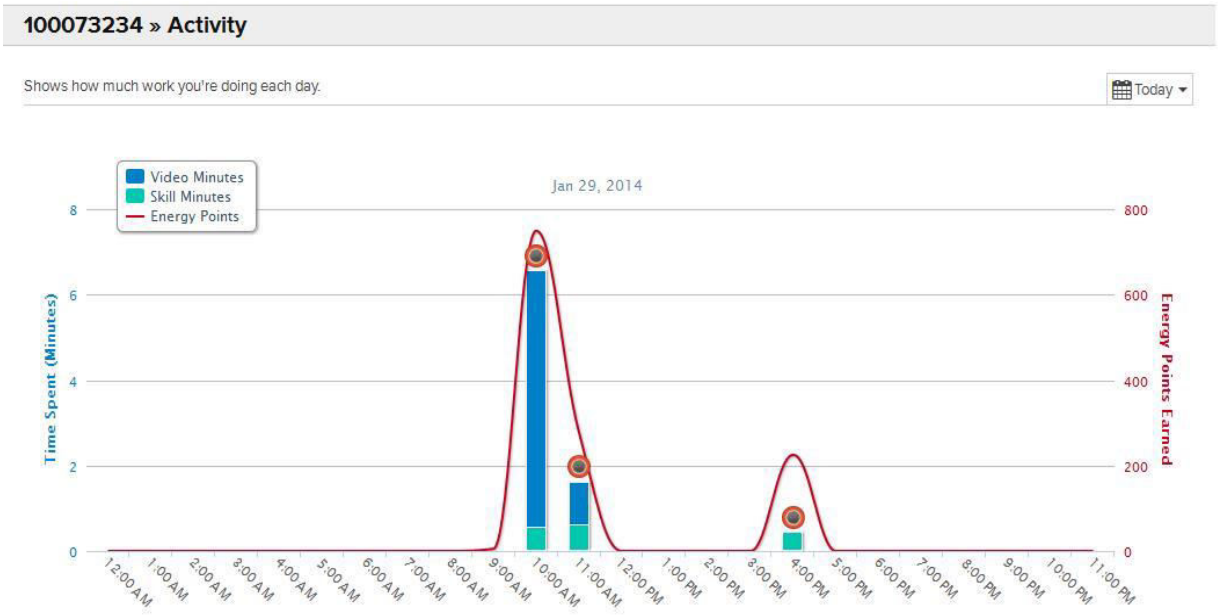

Figura 8. *Informe de activity [3]*

En la figura 9 encontramos otro de los informes disponibles en la plataforma, *Focus*, donde podemos ver la cantidad de tiempo que el usuario ha dedicado a los diferentes temas. En el círculo interior se muestra el tiempo dedicado a cada vídeo, y en el círculo exterior el tiempo dedicado a los ejercicios. Al igual que el informe *Activity*, manteniendo el cursor sobre cualquiera de los círculos, se amplía la información.

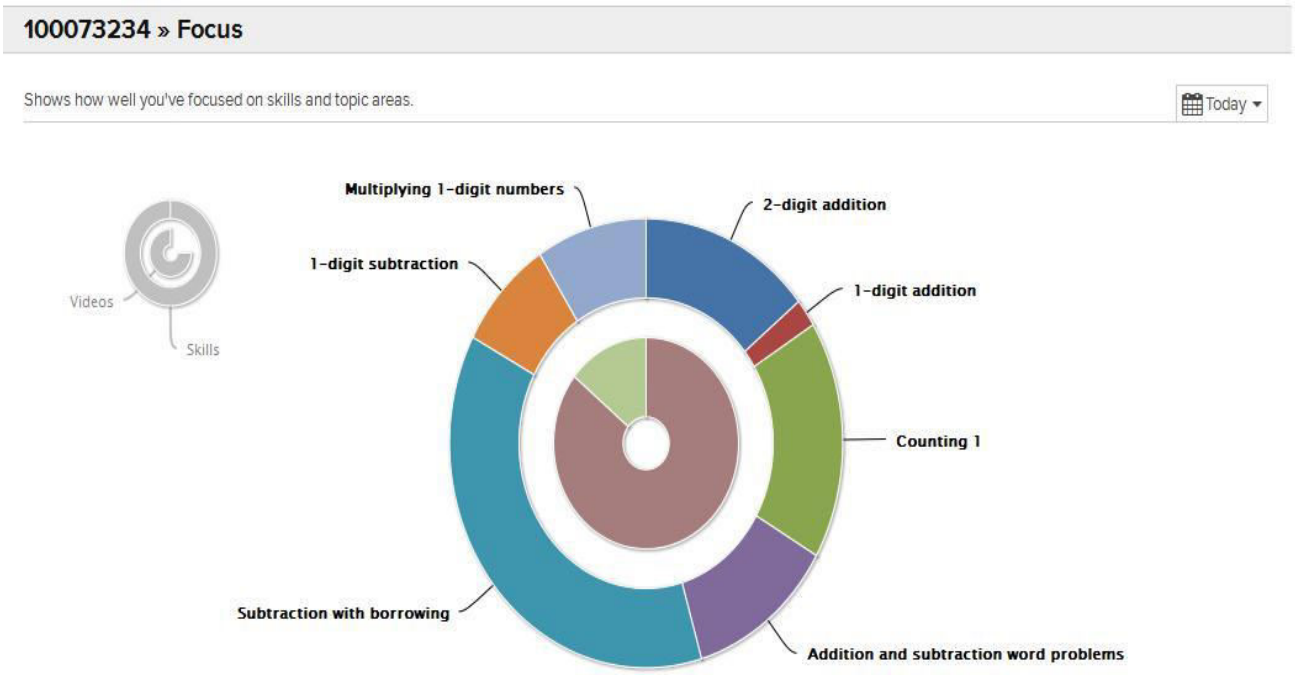

Figura 9. *Visualización Focus [3]*

2.2.3. Módulo ALAS-KA de analítica de aprendizaje

ALAS-KA [8] es un módulo que pretende ampliar la funcionalidad existente en la plataforma Khan Academy con la inclusión de nuevas medidas y visualizaciones sobre el proceso de aprendizaje de los estudiantes. Para mejorar la experiencia de aprendizaje en la plataforma, el objetivo es que las nuevas visualizaciones sean accesibles por los profesores y por los alumnos.

En la figura 10 se muestra el diagrama del sistema ALAS-KA integrado con Khan Academy en el mismo servidor de Google App Engine.

El procesado de datos y la generación de visualizaciones por parte de ALAS-KA no debe ser un hecho aislado en el tiempo, sino que debe soportar el poder actualizarse continuamente en el tiempo de forma periódica, para que las visualizaciones siempre estén actualizadas. Estas actualizaciones se ejecutarán en segundo plano, siendo invisibles para el usuario. Más adelante se explicará cómo se ha logrado esto.

Dado que todo el trabajo ha sido realizado sobre una versión off-line de la Khan Academy, ha habido un período de integración del módulo de detección de sentimientos en ALAS-KA. Esta integración fue realizada junto con un compañero que desarrolló el módulo ALAS-KA, y que está referenciado al comienzo de este apartado.

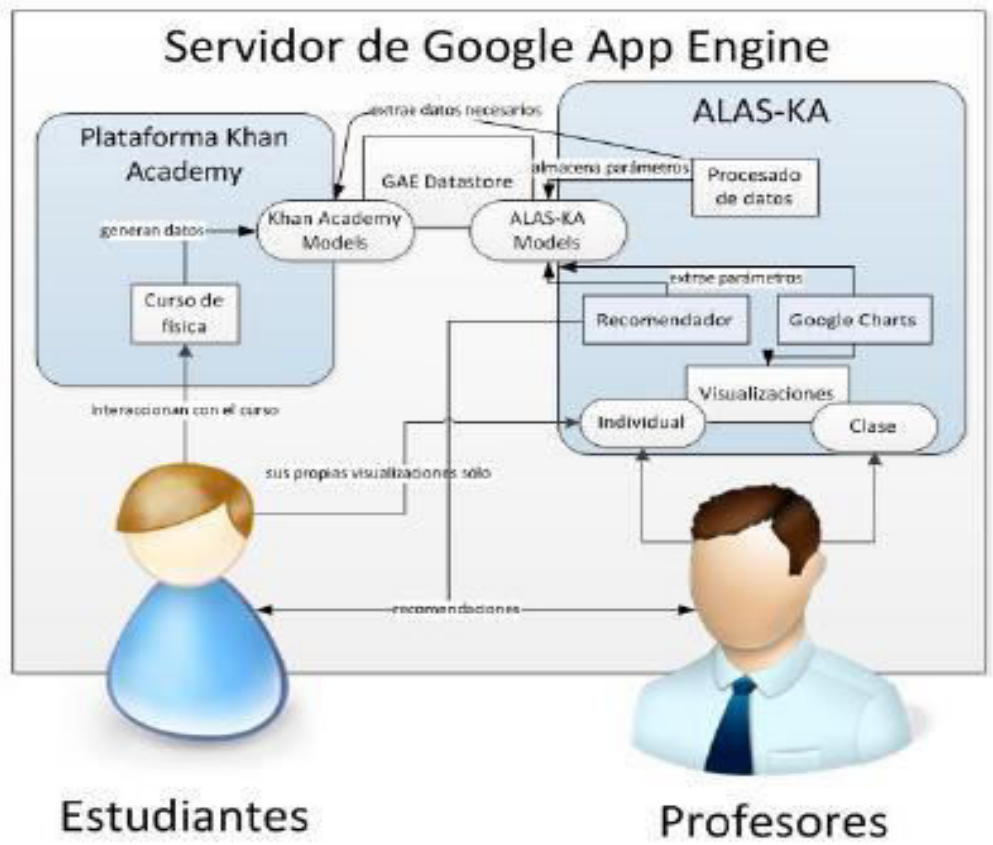

Figura 10. *Diagrama del sistema ALAS-KA integrado con la plataforma Khan Academy [2]*

2.3. Reconocimiento de emociones

En los sistemas computacionales cada vez es más demandada la inclusión de las características afectivas. Esta temática engloba dos grandes áreas: la síntesis de emociones por parte de un sistema computacional y el reconocimiento automático de las emociones expresadas por una persona [9]. Para el trabajo realizado en este proyecto nos interesa la segunda temática.

En [10] se realiza un estudio del reconocimiento de emociones de los estudiantes durante la actividad de desarrollo de software. En este estudio se utilizan los datos generados durante la activad para realizar un modelo de Markov para cada emoción: alegría, frustración, aburrimiento y confusión. Los estudiantes pasan por diferentes estados afectivos que afectan a su rendimiento, por ejemplo, no encontrar una solución a un fallo de compilación o no comprender un mensaje de error.

2.3.1. Reconocimiento de emociones mediante sensores

Ha habido muchos estudios que se han centrado en los factores fisiológicos y en cómo éstos afectan al estado emocional de una persona. Para la detección de

emociones mediante sensores el estudio se ha enfocado a los factores fisiológicos y su relación con los estados afectivos de la persona. Estos estudios hacen uso de sensores corporales para la medición de características como ritmo cardíaco, actividad eléctrica del corazón, conductividad de la piel y actividad bioeléctrica cerebral. La desventaja del uso de sensores es la dificultad del usuario para olvidarse de ellos. Esto puede alterar los resultados al influir en la concentración de la persona.

En [11] se hace una investigación con dos objetivos: 1) examinar sistemáticamente la relación entre el estado afectivo de los estudiantes y los resultados deseados y 2) evaluar la importancia y viabilidad de trazar los estados emocionales de los estudiantes dentro de una situación real. En esta investigación sensores fisiológicos como una cámara para las expresiones faciales, un ratón con sensibilidad a la presión aplicada por los dedos, una silla especial para la detección de la postura del alumno y un sensor en la muñeca para captura la conductividad de la piel. Las emociones a determinar en este trabajo son la confianza, la frustración, el entusiasmo y el interés, concluyendo que la cámara y la silla son los sensores con mayor correlación con el estado de interés.

El estudio realizado en [12] y [13] quería averiguar cuáles son los estados afectivos más frecuentes durante una actividad de aprendizaje. La conclusión fue que las cuatro emociones más frecuentes son frustración, aburrimiento, confusión y entusiasmo, las cuales son casi las mismas que las desarrolladas en este proyecto. Para la detección de estas emociones se utilizaron sensores y se incluyó un análisis conversacional en el ITS (sistema inteligente de enseñanza) y el alumno.

2.3.2. Reconocimiento de emociones mediante la captura de eventos

Como antes mencionamos en [10] se realiza un estudio de la detección de sentimientos mediante la captura de eventos en el desarrollo de software. Los eventos capturados son bastante diversos, como el resultado de una compilación, la generación de ficheros o el uso del navegador web. En este artículo se propuso el análisis de eventos para obtener información afectiva a través de modelos ocultos de Markov (HMM). Este modelo posee la ventaja de ser invisible para el usuario, por lo que no infiere en su concentración.

Este modelo hace uso de una máquina virtual para la realización de una asignatura concreta. Esta máquina virtual guarda la interacción de los estudiantes con un conjunto de aplicaciones necesarias para el dominio de la asignatura, y almacena todos los datos en un fichero o *log*. Además de los datos generados por la interacción en los *log* también se almacenan eventos generados de fuentes externas como el acceso a la página de dicha asignatura. Por último también se le pide al estudiante, de forma recurrente, que indique información explícita sobre su estado anímico. En la figura 11 se muestra como el estudiante puede cambiar los valores de lo siguientes

sentimientos: alegría, aburrimiento, confusión y frustración. Para entrar en más detalle en el proceso de captura de eventos se puede consultar [14] y para experiencias relacionadas con dichas capturas [15].

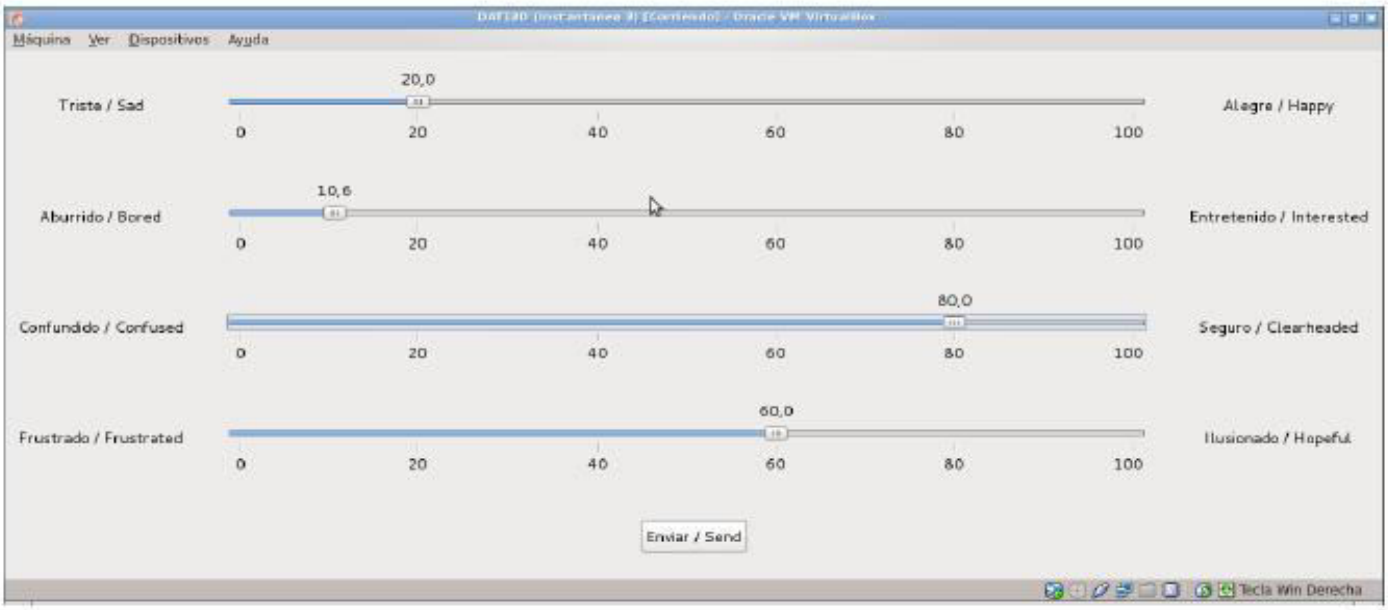

Figura 11. *Formulario sobre el estado anímico del estudiante [10]*

Para el uso del modelo de Markov (HMM) se definen cinco estados: trabajo en actividad (E_1), encuentro de dificultad (E_2), búsqueda de información (E_3), solución a dificultad (E_4) y distracción (E_5). También se definen un listado de símbolos (tabla 1) y las probabilidades asignadas inicialmente de cada estado emita un símbolo (tabla 2).

Observación	Descripción	Herramientas que la generan
command	Instrucción introducida en la línea de comandos	Bash
forum	Cualquier interacción con el foro de la asignatura: acceder a un hilo, crear un nuevo hilo y enviar un mensaje.	Foros en LMS Moodle
compile-error	Compilación con resultado erróneo	GCC, Java
compile-ok	Compilación con resultado exitoso	GCC, Java
debugger	Interacción con la herramienta de depuración	GDB
ide	Interacción con el entorno de desarrollo	KDevelop
lms	Interacción con herramientas del ambiente virtual de aprendizaje que no sean el foro	Moodle
text-editor	Arranque o finalización del editor de texto	Kate
resource-external	Acceso a material no relacionado a la asignatura	Google Chrome, Mozilla Firefox
resource-internal	Acceso a material relacionado con la asignatura (basado en la URL del material)	Google Chrome, Mozilla Firefox
resource-search	Acceso al motor de búsqueda Google	Google Chrome, Mozilla Firefox
memory-ok	Análisis de manejo de memoria con resultado exitoso	Valgrind
memory-error	Análisis de manejo de memoria con resultado erróneo	Valgrind

Tabla 1. *Listado de símbolos [10]*

Símbolo	E_1	E_2	E_3	E_4	E_5
command	0.25	0.05	0.00	0.05	0.00
forum	0.10	0.00	0.30	0.00	0.00
compile-error	0.15	0.70	0.00	0.00	0.00
compile-ok	0.10	0.00	0.00	0.50	0.00
debugger	0.05	0.00	0.00	0.05	0.00
ide	0.05	0.00	0.00	0.05	0.00
lms	0.10	0.00	0.10	0.00	0.05
text-editor	0.10	0.00	0.00	0.00	0.00
resource-external	0.00	0.00	0.10	0.00	0.95
resource-internal	0.00	0.00	0.30	0.00	0.00
resource-search	0.00	0.00	0.20	0.00	0.00
memory-ok	0.05	0.00	0.00	0.35	0.00
memory-error	0.05	0.25	0.00	0.00	0.00

Tabla 2. *Prob. de cada estado de emitir un símbolo [10]*

Como podemos observar, las probabilidades de emisión de símbolos se mantienen sin importar el estado en el que te encuentres. Ahora bien, esto no se puede aplicar a las probabilidades de transición entre estados, así que se definen las probabilidades de transición para cada una de las emociones (tabla 3, tabla 4, tabla 5 y tabla 6). Los estados anímicos de confusión y aburrimiento suelen clasificarse como estados cognitivos, pero se ha observado que tienen un nivel alto de influencia en la ganancia de aprendizaje [16].

Estado	E_1	E_2	E_3	E_4	E_5
E_1	0.60	0.10	0.10	0.10	0.10
E_2	0.30	0.20	0.30	0.10	0.10
E_3	0.40	0.10	0.20	0.20	0.10
E_4	0.60	0.10	0.10	0.10	0.10
E_5	0.60	0.10	0.10	0.10	0.10

Tabla 3. *Probabilidad de transición para el estado de alegría [10]*

Estado	E_1	E_2	E_3	E_4	E_5
E_1	0.40	0.30	0.10	0.10	0.10
E_2	0.10	0.50	0.20	0.05	0.15
E_3	0.25	0.50	0.10	0.05	0.10
E_4	0.70	0.15	0.05	0.05	0.05
E_5	0.30	0.30	0.15	0.05	0.20

Tabla 4. *Probabilidad de transición para el estado de frustración [10]*

Estado	E_1	E_2	E_3	E_4	E_5
E_1	0.40	0.30	0.05	0.05	0.20
E_2	0.15	0.50	0.15	0.05	0.15
E_3	0.20	0.30	0.20	0.10	0.20
E_4	0.60	0.20	0.05	0.05	0.10
E_5	0.20	0.30	0.05	0.05	0.40

Tabla 5. *Probabilidad de transición para el estado de confusión [10]*

Estado	E_1	E_2	E_3	E_4	E_5
E_1	0.40	0.10	0.10	0.10	0.30
E_2	0.10	0.20	0.20	0.10	0.40
E_3	0.15	0.20	0.20	0.15	0.30
E_4	0.40	0.10	0.05	0.05	0.40
E_5	0.20	0.10	0.05	0.05	0.60

Tabla 6. *Probabilidad de transición para el estado de aburrimiento [10]*

En la tabla 7 se muestra el resultado para una secuencia de símbolos de 20 eventos. Cada emoción está calculada como el logaritmo de la probabilidad de que la secuencia de eventos previa sea producida por el HMM del estado anímico.

No.	Símbolo	Alegría	Frustración	Confusión	Aburrimiento
1	command	**-53.75**	-62.49	-58.41	-57.30
2	resource-internal	**-54.56**	-62.80	-59.28	-57.86
3	resource-internal	**-54.75**	-63.35	-59.09	-57.82
4	compile-error	**-54.12**	-61.33	-57.41	-56.69
5	compile-error	**-53.02**	-59.23	-55.09	-55.39
6	compile-error	**-52.46**	-57.50	-53.49	-54.68
7	resource-search	**-53.30**	-58.13	-54.52	-55.48
8	resource-search	**-53.74**	-58.92	-54.49	-55.55
9	resource-external	-53.70	-58.57	**-53.54**	-54.28
10	compile-error	-52.92	-56.92	**-51.90**	-53.65
11	resource-search	-53.38	-57.17	**-52.22**	-53.96
12	resource-search	-53.25	-57.49	**-51.85**	-53.76
13	compile-error	-52.48	-54.93	**-50.46**	-52.76
14	compile-error	-51.42	-52.46	**-48.67**	-51.83
15	compile-ok	-50.99	-52.22	**-49.03**	-51.77
16	debugger	-51.53	-52.11	**-49.78**	-52.72
17	debugger	-52.07	-52.24	**-50.72**	-53.61
18	compile-ok	-52.47	-51.13	**-50.61**	-53.26
19	compile-error	-49.80	-49.23	**-49.11**	-52.63
20	compile-ok	**-50.38**	-51.22	-51.19	-53.36
21	debugger	**-51.93**	-53.63	-53.80	-55.23
22	memory-ok	**-53.17**	-54.67	-54.88	-56.32
23	memory-error	**-52.85**	-53.27	-54.09	-56.20
24	compile-ok	**-51.36**	-52.27	-53.92	-55.94
25	memory-error	**-52.29**	-53.68	-55.19	-56.96
26	text-editor	**-53.87**	-55.68	-56.81	-58.85
27	compile-ok	**-52.85**	-54.18	-56.32	-58.04
28	text-editor	**-52.17**	-54.66	-56.50	-57.99

Tabla 7. *Resultado de símbolo generados y el logaritmo de la probabilidad para cada estado anímico [10]*

Este estudio concluye que es viable este modelo para la detección de emociones, además de poder extender el modelo para determinar más estados anímicos.

2.4. Tecnologías utilizadas

Para la implementación de este módulo de detección de sentimientos se han utilizado las tecnologías ahora descritas. Mencionar que se han tenido que aprender todas las tecnologías excepto HTML, de la cual se tenían conocimientos y solo tuvo que repasar.

2.4.1. Python

Python es un lenguaje de programación ampliamente utilizado de alto nivel, con una sintaxis muy sencilla y que favorece un código legible. Se trata de un lenguaje interpretado o de script, fuertemente tipado, con tipado dinámico, multiplataforma y orientado a objetos. La versión utilizada para el desarrollo del módulo ha sido python 2.7 [17].

***Figura 12**. Python*

2.4.2. HTML

HTLM [18] es un lenguaje para la definición de contenido de una página web. Basa su filosofía de desarrollo en la referenciación, de esta manera la página web solo contiene texto, mientras que para añadir una imagen, script, vídeo, etc. Se hace una referencia a la ubicación.

2.4.3. JavaScript

JavaScript [19] es un lenguaje de programación interpretado, orientado a objetos, débilmente tipado y dinámico. Su uso más frecuente viene dado como parte de un navegador web mejorando la interfaz del usuario. Normalmente se utiliza junto con HTML.

2.4.4. Google App Engine

Google App Engine [20] es una infraestructura de Google que te permite ejecutar tus aplicaciones web. Estas aplicaciones son de fácil creación, mantenimiento y ampliación conforme el tráfico aumenta y el almacenamiento de datos crece. Puedes compartir tu aplicación con todas las personas o limitar el acceso.

Google App Engine admite dos lenguajes de programación, Python y Java, y en estos momentos se encuentra en fase de experimentación con Go y PHP. El uso de esta infraestructura es gratuito hasta que no se excedan unos límites. Entre otras funciones, Google App Engine incluye:

- Servidor web dinámico

- Almacenamiento permanente
- Escalado automático y distribución de carga
- Tareas programadas para activar eventos en momentos determinados y en intervalos regulares.

Figura 13. *Google App Engine*

2.4.5. App Engine Datastore

El App Engine Datastore [21] es una base de datos de tipo no relacional que pone a nuestra disposición un almacenamiento sólido y escalable. Proporciona un tipo de lenguaje parecido a SQL llamado GQL (Google Query Language). La función de este almacén de datos es la de guardar objetos de datos, conocidos con el nombre de *entities* o entidades. Cada entidad puede contener una o más propiedades que pueden ser de cualquiera de los tipos de datos admitidos y contiene una *key* o clave que sirve de identificador único.

En Python, una clase de este tipo crea en el almacén de datos una entidad a través de una llama a *model*. Esta entidad creada va a tener como nombre el nombre de la clase Python y adquirirá las propiedades que tenga la clase Python como atributos. Para crear una entidad se necesita invocar al constructor de la clase y posteriormente almacenarla haciendo uso de la función *put()*.

En el almacén de datos de App Engine, todos los intentos por crear, eliminar o modificar una entidad ocurren en una transacción, garantizando que, si ocurre un error, no se aplique los cambios. En una misma transacción se pueden realizar cambios en varias entidades mediante los grupos de entidades.

El almacén de datos de App Engine tiene diferencias importantes con una base de datos relacional:

- Es escalable, con los que un aumento del nivel de tráfico no implica que el nivel de rendimiento de una aplicación disminuya.

- Entidades del mismo tipo pueden tener propiedades diferentes, o que distintas entidades puedan tener propiedades con el mismo nombre pero sean de diferente tipo.

- Los tipo de consulta son más limitados que en una base de datos tradicional, porque las consultas que se realizan las proporcionan índices creados previamente.

2.4.6. Google Charts

Google Charts [22] es una herramienta que permite a los desarrolladores de aplicaciones web crear visualizaciones. La manera más frecuente de utilizar es mediante un JavaScript incrustado en tu página web. Para hacerlo necesitas cargar las librerías, enumerar los datos que queremos registrar en las visualizaciones, elegir las opciones de personalización y crear un objeto chart con un *id*.

Google Charts dispone de una amplia galería de visualizaciones, desde sencillas barras, a dibujos por áreas, círculos o árboles. Para el trabajo que se ha realizado se ha elegido representar los sentimientos en visualizaciones lineales.

2.5. Versión off-line de la plataforma Khan Academy

Durante todo el desarrollo del módulo para la detección de sentimientos, se ha trabajado sobre una versión off-line de la plataforma Khan Academy. En la figura 14 se muestra una imagen de esta versión de la plataforma.

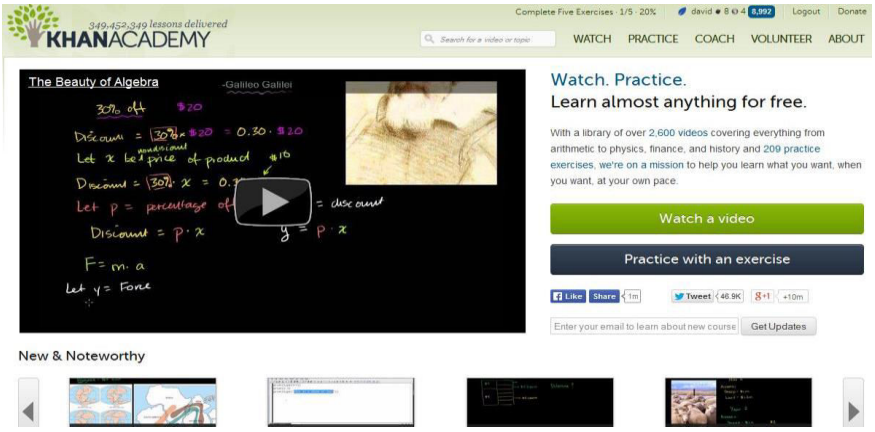

Figura 14. *Versión off-line de la plataforma Khan Academy*

Esta versión off-line trae una serie de entidades ya creadas en la base datos, a las que hay que sumar las creadas durante el desarrollo del trabajo, pero esto se explicará más adelante. Ahora se va a hacer un repaso a las entidades que ya venían en esta versión, las cuales han sido de mayor utilidad:

- UserData: Esta entidad guarda información general sobre los usuarios registrados, conteniendo el nombre, el usuario, los vídeos vistos, la fecha en la que interactuó por última vez con la plataforma, los puntos ganados, etc.

- UserExercise: Esta entidad nos proporciona la relación de un usuario con un ejercicio específico. Nos da información de si un usuario ha hecho bien un ejercicio, cuántas veces ha accedido a dicho ejercicio, etc.

- UserVideo: Esta entidad tiene la misma funcionalidad que la anterior, relaciona un usuario con un vídeo específico. Entre otras cosas, nos proporciona si un usuario ha visto un vídeo completo, la cantidad de segundos vistos de un vídeo, etc.

- Exercise y Video: Estas entidades contienen todos los ejercicios y vídeos disponibles en la Khan Academy. La información más relevante que proporcionan es el nombre, la fecha de creación, etc.

- UserBadge: Esta entidad guarda la relación entre un usuario y los badges ganados. De la información que nos proporciona cabe destacar que, de cada usuario, nos dice los badges que ha ganado, la fecha y hora exacta en la que los ganó y el ejercicio en el cual ganó cada badge.

- ProblemLog: Esta entidad nos proporciona información cuando un usuario accede a un ejercicio (*ProblemLog*). Cada vez que un usuario accede a un ejercicio y da una respuesta, esta entidad guarda un *ProblemLog* con mucha información, donde la que resulta más útil es la marca de tiempo y si la respuesta ha sido correcta o no. De esta manera un ejercicio puede aparecer en distintos *ProblemLog*, dependiendo del número de respuestas, si ha sido correcto o no, etc. Dado que esta entidad es la más importante para el desarrollo del módulo de detección de sentimientos, se van a mostrar en la figura 15 diez ProblemLog y se va a explicar en mayor detalle la información más relevante.

  1- Esta es la *Key Name*. De aquí se puede deducir el nombre del usuario, el nombre del ejercicio (que también se ve en 5) y el número de intento del ejercicio. La imagen nos muestra el tercer y cuarto intento en el ejercicio *addition_2* del usuario *manolo* y el primer y único intento del ejercicio *addition_3*.

  2- Nos muestra lo mismo que 1 pero esta vez para el usuario *nuria*. Nos dice que ha intentado el ejercicio *addition_1* seis veces y el ejercicio *addition_3* una vez.

  3- Este dato es muy importante pues nos deja la marca de tiempo en la cual ocurre el ejercicio. Será necesario a la hora de delimitar qué rango de tiempo queremos coger para mostrar la visualización de los sentimientos.

  4- Otro dato de los más importantes a la hora de decidir que sentimiento predomina sobre los demás. Nos dice si la respuesta a un ejercicio se ha resuelto de forma correcta o de forma errónea.

5- Como ya se ha mencionado en 1, es el nombre del ejercicio que se ha intentado resolver.

Key Name	attempts	backup_timestamp	correct	count_attempts	count_hints	earned_proficiency	exercise	time_done
problemlog_manolo@example.com_addition_2_3	[u""97""]	2014-01-08 09:59:21	True	1	0	False	addition_2	2014-01-08 09:59:19
problemlog_manolo@example.com_addition_2_4	[u""13"", u""14"", u...	2014-01-08 09:59:33	False	3	0	False	addition_2	2014-01-08 09:59:23
problemlog_manolo@example.com_addition_3_1	[u""121"", u""988"",...	2014-01-08 10:00:33	False	9	0	False	addition_3	2014-01-08 09:59:49
problemlog_nuria@example.com_addition_1_1	[u""19""]	2013-10-28 16:38:14	True	1	0	False	addition_1	2013-10-28 16:38:11
problemlog_nuria@example.com_addition_1_2	[u""13""]	2013-10-28 16:38:15	True	1	0	False	addition_1	2013-10-28 16:38:13
problemlog_nuria@example.com_addition_1_3	[u""13""]	2013-10-28 16:38:18	True	1	0	False	addition_1	2013-10-28 16:38:16
problemlog_nuria@example.com_addition_1_4	[u""9""]	2013-10-28 16:38:19	True	1	0	False	addition_1	2013-10-28 16:38:17
problemlog_nuria@example.com_addition_1_5	[u""11""]	2013-10-28 16:38:22	True	1	0	True	addition_1	2013-10-28 16:38:19
problemlog_nuria@example.com_addition_1_6	[u""3""]	2013-10-28 16:38:24	True	1	0	False	addition_1	2013-10-28 16:38:22
problemlog_nuria@example.com_addition_3_1	[u""890""]	2013-10-28 16:38:38	True	1	0	False	addition_3	2013-10-28 16:38:37

‹ Previous 29 30 31 32 33 34 35

Figura 15. *Visualización de la entidad ProblemLog*

Para poder obtener datos de bajo nivel a través de esta versión off-line, se han tenido que crear usuarios, pues trae los ejercicios y los vídeos, pero no contiene ningún dato de la interacción de usuarios con la plataforma.

El procedimiento para ello fue crear distintos usuarios, e interactuar con la plataforma como si tuvieran diferentes *skills* o habilidades. De esta manera, a unos usuarios se les daría mejor un tema que a otros, unos fallarían más respuestas de forma continuada, y otros fallarían respuestas de forma más intermitente, alternando con aciertos. Lo que se pretendía era poder tener delante cualquier caso que pudiera suceder en la plataforma on-line de la Khan Academy.

Para pode trabajar sobre esta versión off-line, es necesario ejecutar un fichero con extensión *.bat*, el cual pone a tu disposición la plataforma en el navegador con la siguiente url : ***http://localhost:8080/***. El resultado de ejecutar esta url se muestra en la figura 14.

Si queremos entrar en la consola como administrador, la url que debemos ejecutar es la siguiente: ***http://localhost:8080/_ah/admin***.

Figura 16. *Consola de administrador*

Como podemos ver en la figura 16, entrar como administrador nos da la opción de ver cualquiera de las entidades, utilizar la consola interactiva (figura 17), o ver los *cron job* y las *task queue* entre otras cosas.

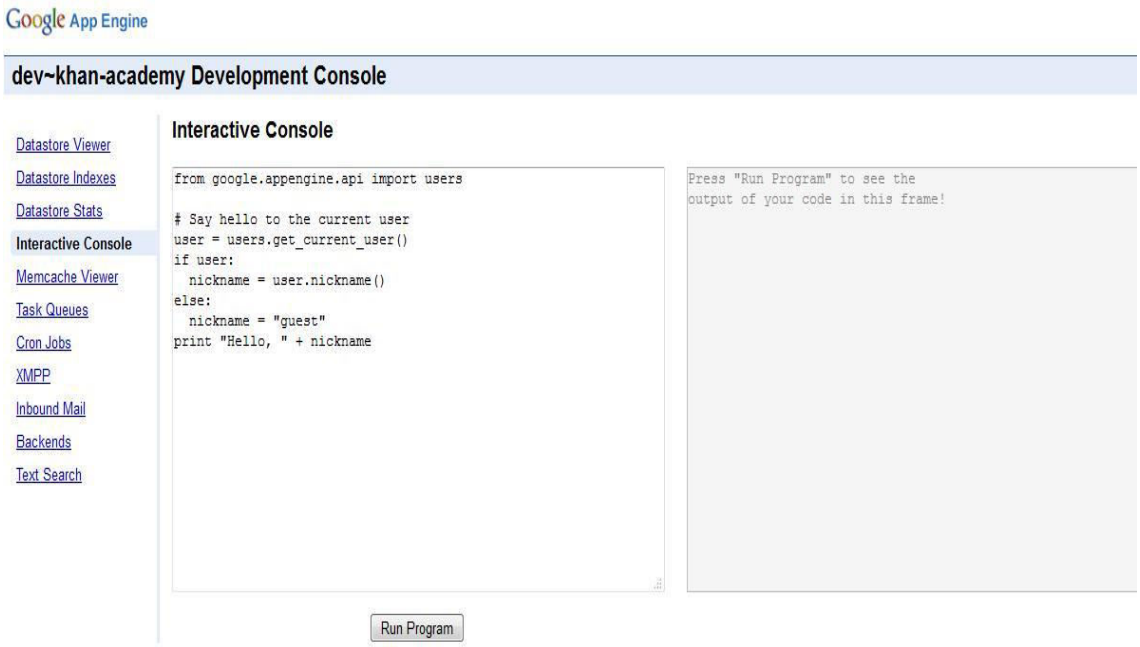

Figura 17. *Consola interactiva*

CAPÍTULO 3

MODELOS DE DETECCIÓN DE SENTIMIENTOS EN LA PLATAFORMA KHAN ACADEMY

Para la detección de sentimientos en la plataforma Khan Academy, únicamente se han recogido los ejercicios realizados por los usuarios y la ganancia de badges como consecuencia de la resolución de ejercicios, no teniendo en cuenta la visualización de vídeos, la personalización del perfil, etc.

Los modelos de detección que se muestran a continuación son parte de la tesis dirigida por Derick Leony, la cual está dirigida por Pedro José Muñoz Merino y Abelardo Pardo. Estos modelos han sido propuestos y hechos por Derick Leony, y en los cuales he colaborado para refinarlos.

3.1. Modelo para la detección de frustración

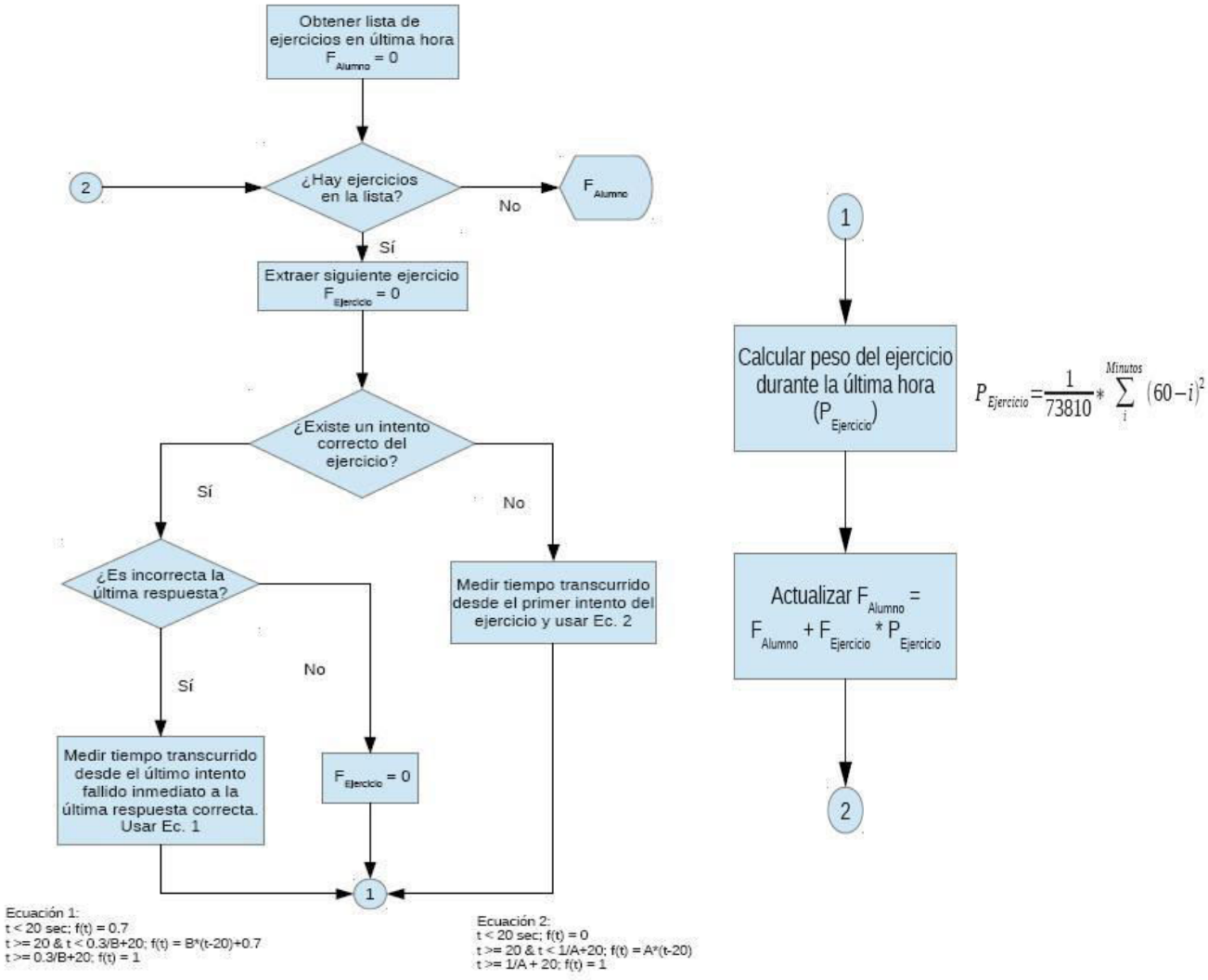

Figura 18. *Diagrama de flujo para la detección de frustración*

En la figura 18 se muestra el organigrama seguido para la implementación de la detección de frustración.

Palabras clave del diagrama de flujo:

- F_{Alumno} : frustración total del alumno
- $F_{Ejercicio}$: frustración de un ejercicio concreto
- $P_{Ejercicio}$: peso asignado a un ejercicio concreto

- Lo primero es obtener todos los ejercicios realizados por un alumno desde 60 minutos antes de la hora actual en la que se ejecuta el proceso. O lo que es lo mismo, obtener todos los *ProblemLog* almacenados en la última hora por un alumno en la entidad *ProblemLog* y guardarlos en una lista que vamos a denominar L_1. Se inicializa F_{Alumno} con el valor cero.

- Se comprueba si L_1 contiene ejercicios.

    - Si L_1 no contiene ejercicios, se guarda en el alumno sobre el que se esté actuando el valor de F_{Alumno} y aquí se termina el proceso de detección de frustración.

    - Si en L_1 hay uno o más ejercicios, se cogen todos los intentos de un mismo tipo de ejercicio y se guardan en otra lista que denominaremos L_2.

- Se comprueba si en L_2 algún intento del ejercicio se ha resuelto de manera correcta.

    - Si en L_2 no se ha resuelto ningún ejercicio de manera correcta, se medirá el tiempo transcurrido desde el primer intento del ejercicio hasta la finalización del ejercicio, recurriremos a la Ecuación 2 mostrada en la figura 18, y esto nos devolverá el valor de $F_{Ejercicio}$ que debemos aplicar.

    - Si L_2 si que contiene una solución correcta en alguno de los intentos del ejercicio se actuará de la siguiente forma:

        - Si la solución correcta se ha obtenido en el último intento del ejercicio se establece $F_{Ejercicio}$ con el valor cero.

        - Si, por el contrario, la solución correcta no se ha obtenido en el último intento del ejercicio, se medirá el tiempo transcurrido desde el último intento fallido inmediato al último intento correcto hasta la finalización del ejercicio, recurriremos a la Ecuación 1 mostrada en la figura 18 y el valor de $F_{Ejercicio}$ que nos devuelva será el que utilizemos.

- Se calcula $P_{Ejercicio}$ mediante la fórmula mostrada en la figura 18. Esta fórmula viene a darnos el peso que ha tenido un ejercicio durante la última hora en la que se calcula el sentimiento. Es decir, no tiene el mismo impacto sobre el estado afectivo de un alumno que se hayan dado respuestas erróneas al principio de la hora sobre la que se está trabajando y respuestas acertadas al final, que al revés. Por este motivo cuanto antes se hayan realizado los intentos de un ejercicio menos peso tendrán en el estado afectivo y viceversa.

- Por último se borra de L_1 el ejercicio procesado, se calcula la F_{Alumno} con la fórmula del último paso que se muestra en la figura 18 y se vuelve a comprobar si en L_1 todavía quedan ejercicios por procesar. Si es así, se vuelve a realizar el mismo proceso que se ha descrito. Si por el contrario no quedan ejercicios, se guarda sobre el alumno que se esté actuando el valor de F_{Alumno}, y ese es el nivel de frustración de un alumno.

3.2. Modelo para la detección de confusión

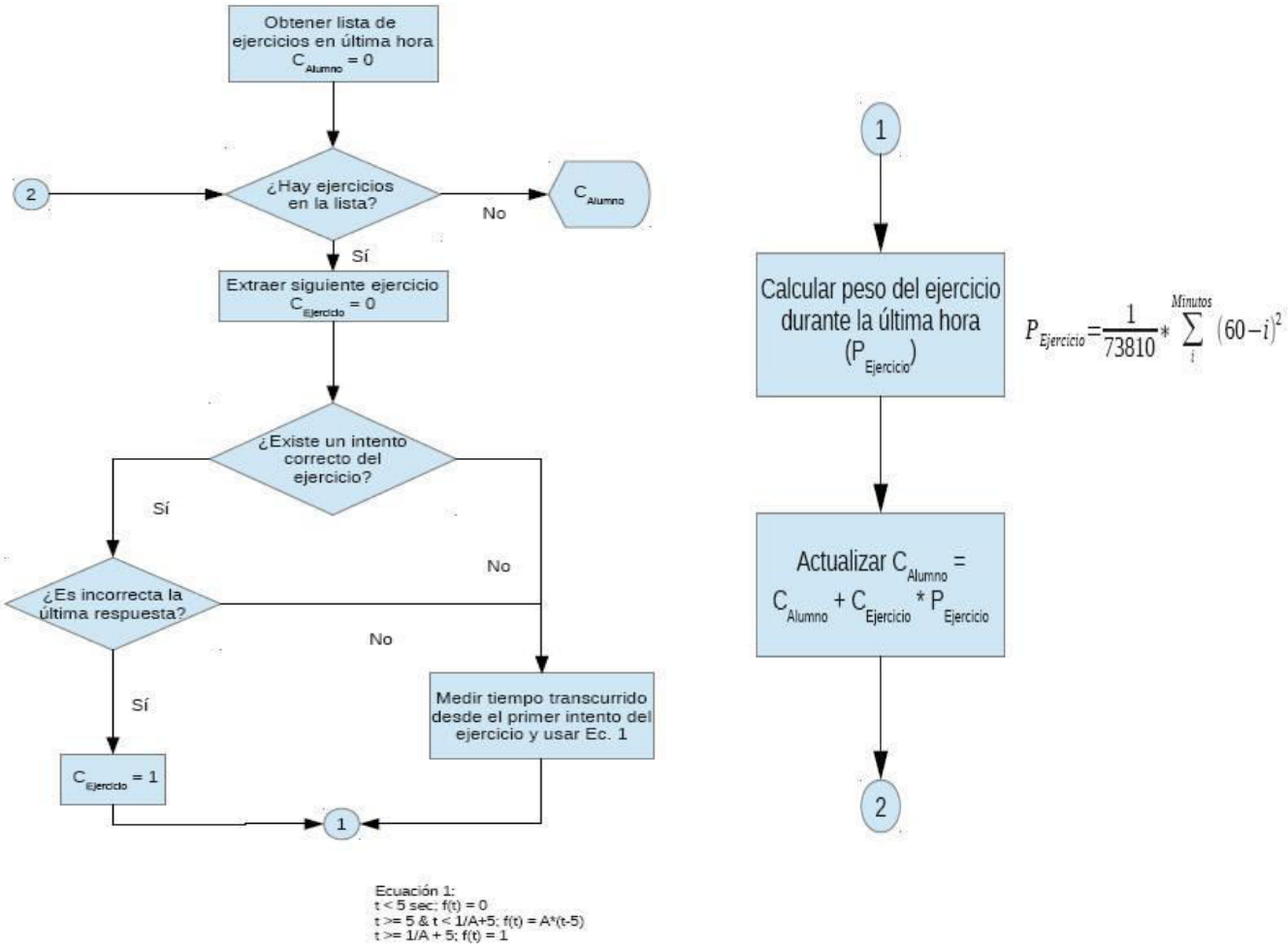

Figura 19. *Diagrama de flujo para la detección de confusión*

En la figura 19 se muestra el organigrama seguido para la implementación de la detección de confusión.

Palabras clave del diagrama de flujo:

- C_{Alumno} : confusión total del alumno
- $C_{Ejercicio}$: confusión de un ejercicio concreto
- $P_{Ejercicio}$: peso asignado a un ejercicio concreto

- Lo primero es obtener todos los ejercicios realizados por un alumno desde 60 minutos antes de la hora actual en la que se ejecuta el proceso. O lo que es lo mismo, obtener todos los *ProblemLog* almacenados en la última hora por un alumno en la entidad *ProblemLog* y guardarlos en una lista que vamos a denominar L_1. Se inicializa C_{Alumno} con el valor cero.

- Se comprueba si L_1 contiene ejercicios.

    - Si L_1 no contiene ejercicios, se guarda en el alumno sobre el que se esté actuando el valor de C_{Alumno} y aquí se termina el proceso de detección de confusión.

    - Si en L_1 hay uno o más ejercicios, se cogen todos los intentos de un mismo tipo de ejercicio y se guardan en otra lista que denominaremos L_2.

- Se comprueba si en L_2 algún intento del ejercicio se ha resuelto de manera correcta.

    - Si en L_2 no existe ninguna solución correcta en ningún intento del ejercicio, se medirá el tiempo transcurrido desde el primer intento del ejercicio hasta el final del ejercicio, se utilizará la Ecuación 1 que se muestra en la figura 19, y el valor de $C_{Ejercicio}$ será el que debemos utilizar más adelante.

    - Si en L_2 si existe algún intento con una solución correcta, se actuará de la siguiente manera:

        - Si la solución correcta se ha obtenido en el último intento del ejercicio, se medirá el tiempo transcurrido desde el primer intento del ejercicio hasta el final del ejercicio, se utilizará la Ecuación 1 que se muestra en la figura 19, y el valor de $C_{Ejercicio}$ será el que debemos utilizar más adelante.

        - Si, por el contrario, la respuesta al último intento del ejercicio no es correcta, se establece el valor de $C_{Ejercicio}$ con el valor uno.

- Se calcula $P_{Ejercicio}$ mediante la fórmula mostrada en la figura 19. Esta fórmula viene a darnos el peso que ha tenido un ejercicio durante la última hora en la que se calcula el sentimiento. Es decir, no tiene el mismo impacto sobre el estado afectivo de un alumno que se hayan dado respuestas erróneas al principio de la hora sobre la que se está trabajando y respuestas acertadas al final, que al revés. Por este motivo cuanto antes se hayan realizado los intentos de un ejercicio menos peso tendrán en el estado afectivo y viceversa.

- Por último se borra de L_1 el ejercicio procesado, se calcula la C_{Alumno} con la fórmula del último paso que se muestra en la figura 19 y se vuelve a comprobar si en L_1 todavía quedan ejercicios por procesar. Si es así, se vuelve a realizar el mismo proceso que se ha descrito. Si por el contrario no quedan ejercicios, se guarda sobre el alumno que se esté actuando el valor de C_{Alumno}, y ese es el nivel de confusión de un alumno.

3.3. Modelo para la detección de aburrimiento

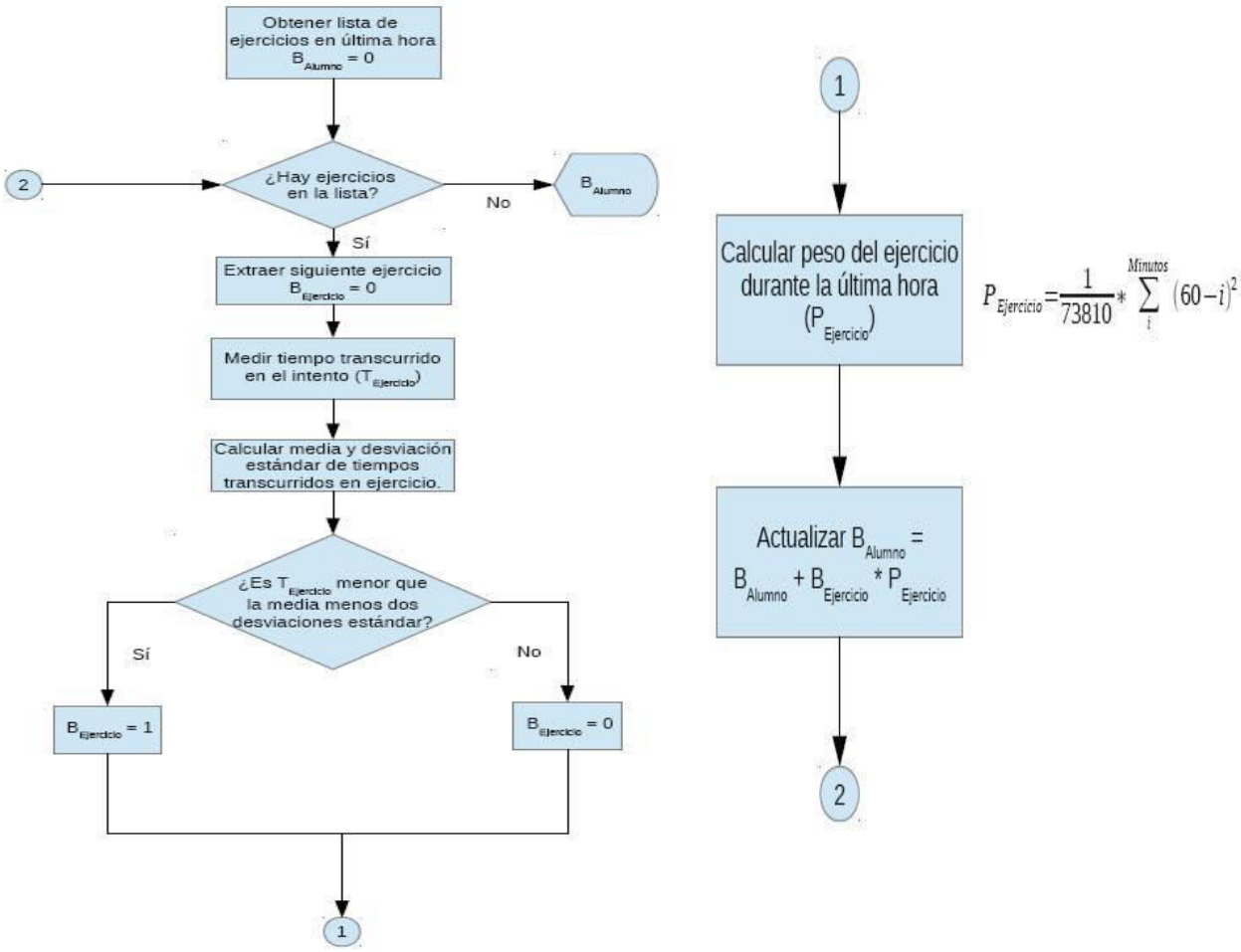

Figura 20. *Diagrama de flujo para la detección de aburrimiento*

En la figura 20 se muestra el organigrama seguido para la implementación de la detección de aburrimiento.

Palabras clave del diagrama de flujo:

- B_{Alumno} : aburrimiento total del alumno
- $B_{Ejercicio}$: aburrimiento de un ejercicio concreto
- $P_{Ejercicio}$: peso asignado a un ejercicio concreto
- $T_{Ejercicio}$: Tiempo total que ha durado un ejercicio

- Lo primero es obtener todos los ejercicios realizados por un alumno desde 60 minutos antes de la hora actual en la que se ejecuta el proceso. O lo que es lo mismo, obtener todos los *ProblemLog* almacenados en la última hora por un alumno en la entidad *ProblemLog* y guardarlos en una lista que vamos a denominar L_1. Se inicializa B_{Alumno} con el valor cero.

- Se comprueba si L_1 contiene ejercicios.

    - Si L_1 no contiene ejercicios, se guarda en el alumno sobre el que se esté actuando el valor de B_{Alumno} y aquí se termina el proceso de detección de aburrimiento.

    - Si en L_1 hay uno o más ejercicios, se cogen todos los intentos de un mismo tipo de ejercicio y se guardan en otra lista que denominaremos L_2.

- Se mide el tiempo que ha durado el ejercicio y se guarda en $T_{Ejercicio}$.

- Se coge el tiempo que ha transcurrido en cada intento del ejercicio y se calcula la media y la desviación estándar de la duración de los intentos.

- Se comprueba si $T_{Ejercicio}$ es menor que la media menos dos desviaciones estándar:

    - En caso afirmativo, se establece el valor de $B_{Ejercicio}$ con el valor uno, y este valor será el que se use más adelante.

    - Por el contrario, si no es afirmativo, se estable el valor de $B_{Ejercicio}$ con el valor cero, y este valor será el que se use más adelante.

- Se calcula $P_{Ejercicio}$ mediante la fórmula mostrada en la figura 20. Esta fórmula viene a darnos el peso que ha tenido un ejercicio durante la última hora en la que se calcula el sentimiento. Es decir, no tiene el mismo impacto sobre el estado afectivo de un alumno que se hayan dado respuestas erróneas al principio de la hora sobre la que se está trabajando y respuestas acertadas al final, que al revés. Por este motivo cuanto antes se hayan realizado los intentos de un ejercicio menos peso tendrán en el estado afectivo y viceversa.

- Por último se borra de L_1 el ejercicio procesado, se calcula la B_{Alumno} con la fórmula del último paso que se muestra en la figura 20 y se vuelve a comprobar si en L_1 todavía quedan ejercicios por procesar. Si es así, se vuelve a realizar el mismo proceso que se ha descrito. Si por el contrario no quedan ejercicios, se guarda sobre el alumno que se esté actuando el valor de B_{Alumno}, y ese es el nivel de aburrimiento de un alumno.

3.4. Modelo para la detección de alegría

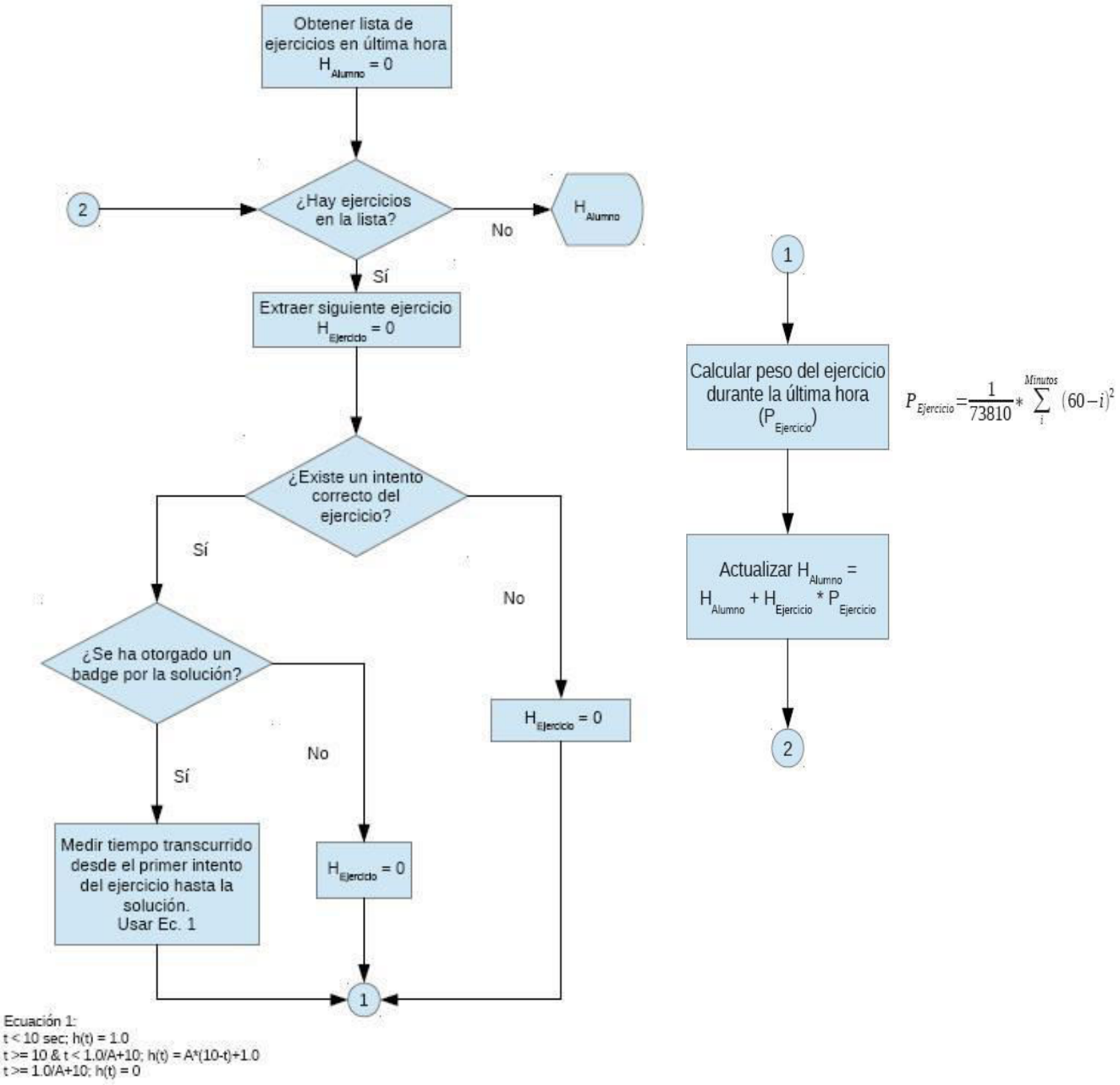

Figura 21. *Diagrama de flujo para la detección de alegría*

En la figura 21 se muestra el organigrama seguido para la implementación de la detección de alegría.

Palabras clave del diagrama de flujo:

- H_{Alumno} : alegría total del alumno
- $H_{Ejercicio}$: alegría de un ejercicio concreto
- $P_{Ejercicio}$: peso asignado a un ejercicio concreto

- Lo primero es obtener todos los ejercicios realizados por un alumno desde 60 minutos antes de la hora actual en la que se ejecuta el proceso. O lo que es lo mismo, obtener todos los *ProblemLog* almacenados en la última hora por un alumno en la entidad *ProblemLog* y guardarlos en una lista que vamos a denominar L_1. Se inicializa H_{Alumno} con el valor cero.

- Se comprueba si L_1 contiene ejercicios.

    - Si L_1 no contiene ejercicios, se guarda en el alumno sobre el que se esté actuando el valor de H_{Alumno} y aquí se termina el proceso de detección de alegría.

    - Si en L_1 hay uno o más ejercicios, se cogen todos los intentos de un mismo tipo de ejercicio y se guardan en otra lista que denominaremos L_2.

- Se comprueba si en L_2 algún intento del ejercicio se ha resuelto de manera correcta.

    - Si en L_2 no existe ninguna solución correcta en ningún intento del ejercicio, se le asignará a $H_{Ejercicio}$ que utilizaremos más adelante

    - Si en L_2 si existe algún intento con una solución correcta, se actuará de la siguiente manera:

        - Si se ha conseguido algún badge durante la resolución del ejercicio, se medirá el tiempo transcurrido desde el principio del ejercicio hasta la solución por la que nos dieron el badge, y se aplicará la Ecuación 1 de la figura 21, y esto nos dará el valor de $H_{Ejercicio}$ que debemos utilizar después

        - Si, por el contrario, no se ha conseguido ningún badge durante la resolución del ejercicio, el valor de $H_{Ejercicio}$ se establecerá a cero.

- Se calcula $P_{Ejercicio}$ mediante la fórmula mostrada en la figura 21. Esta fórmula viene a darnos el peso que ha tenido un ejercicio durante la última hora en la que se calcula el sentimiento. Es decir, no tiene el mismo impacto sobre el estado afectivo de un alumno que se hayan dado respuestas erróneas al

principio de la hora sobre la que se está trabajando y respuestas acertadas al final, que al revés. Por este motivo cuanto antes se hayan realizado los intentos de un ejercicio menos peso tendrán en el estado afectivo y viceversa.

- Por último se borra de L_1 el ejercicio procesado, se calcula la H_{Alumno} con la fórmula del último paso que se muestra en la figura 21 y se vuelve a comprobar si en L_1 todavía quedan ejercicios por procesar. Si es así, se vuelve a realizar el mismo proceso que se ha descrito. Si por el contrario no quedan ejercicios, se guarda sobre el alumno que se esté actuando el valor de H_{Alumno}, y ese es el nivel de alegría de un alumno.

Como se puede comprobar viendo todos los organigramas, para la detección de estos cuatro sentimientos sólo se han tenido en cuenta los ejercicios realizados por los usuarios y los badges ganados con los ejercicios bien resueltos, no se han tenido en cuenta, por ejemplo, los vídeos vistos o los badges ganados personalizando el perfil de usuario, pues se pueden ganar badges de más formas que sólo resolviendo ejercicios.

CAPÍTULO 4

IMPLEMENTACIÓN DEL DETECTOR DE SENTIMIENTOS EN LA PLATAFORMA KHAN ACADEMY

En este capítulo se va a mostrar un diagrama de las clases creadas, se va a explicar cada una estas clases, y se explicarán las visualizaciones con las que se realizaron pruebas antes de la integración en ALAS-KA.

4.1. Diagrama de clases

La figura 22 muestra un diagrama de clases del módulo de detección de sentimientos implementado, el cual se va a explicar por niveles para facilitar su compresión.

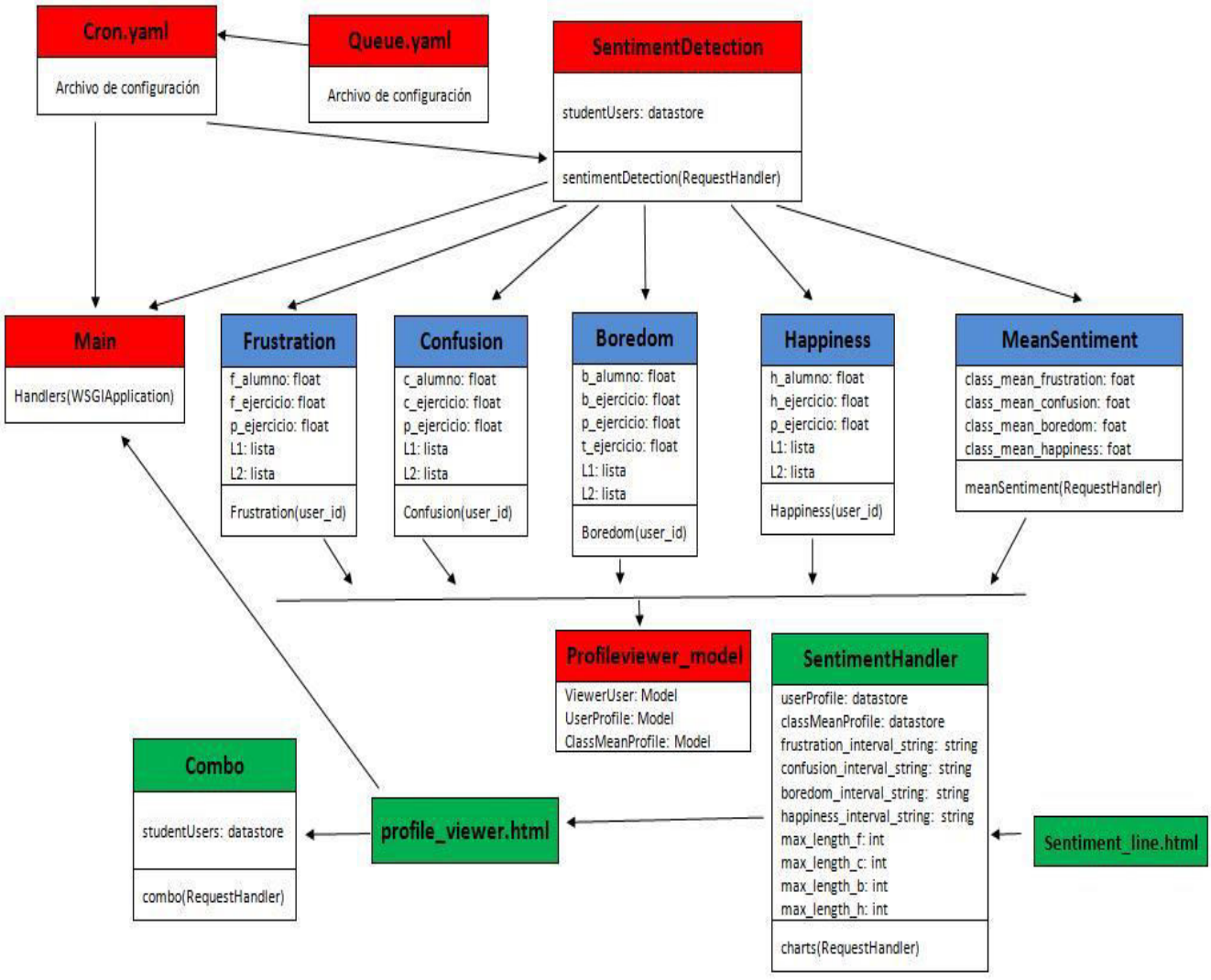

Figura 22. *Diagrama de clases*

Las clases recuadradas en rojo son de primer nivel pues sin ellas no se podrían guardas las medidas (profileviewer_model.py), no se podría ir redirigiendo mediante handlers de un fichero a otro (main.py) y, por supuesto, no se podría configurar el módulo para que de forma periódica actualizara las medidas de los sentimientos (cron.yaml, queue.yaml, SentimentDetection.py).

Las clases recuadradas en azul son clases de segundo nivel pues todas ellas se dedican a calcular el nivel de cada sentimiento de cada alumno por separado (frustration.py, confusion.py, boredom.py, happiness.py) y el nivel medio de todos los sentimientos de toda una clase (MeanSentiment.py).

Las clases recuadradas en verde son de tercer nivel, pues su función es la de hacer posible las visualizaciones. Primero las clases python (Combo.py, SentimentHandler.py) recuperan datos y los ficheros html (profile_vieser.html, sentiment_line.html) muestran las visualizaciones.

4.2. Descripción de clases y archivos de configuración y visualización

cron.yaml: Este es un fichero de configuración de un *cron job* [23]. App Engine tiene un servicio *cron* que te permite configurar tareas programadas de manera que se ejecuten periódicamente o en intervalos regulares. App Engine te permite configurar una o más *tareas cron* (con un máximo de 20 por aplicación), como por ejemplo, el envío de un correo electrónico periódicamente, actualizar datos almacenados en caché, etc. Una *tarea cron* invoca una URL, mediante una solicitud GET HTTP, en un determinado momento del día. Así, se pasa a describir la configuración del *cron* realizado en este trabajo:

```
cron:

- description: Sentiment-Detection
  url: /admin/SentimentDetection
  schedule: every 1 hours
  timezone: Europe/Madrid
```

El campo *description* hace referencia al nombre que se le da al *cron*. En *timezone* se define la zona horario en la que se actúa, sino se especifica, la programación se ejecuta en hora GMT. En *url* se especifica una URL de una aplicación que se ejecuta a través del servicio cron. Y por último en *schedule* se indica la periodicidad con la que quieres que se ejecute el servicio cron, que en este caso se va a ejecutar un total de 24 veces al día. En la figura 23 vemos cómo ha quedado configurado el *cron job* creado (recuadrado de amarillo) en la plataforma off-line.

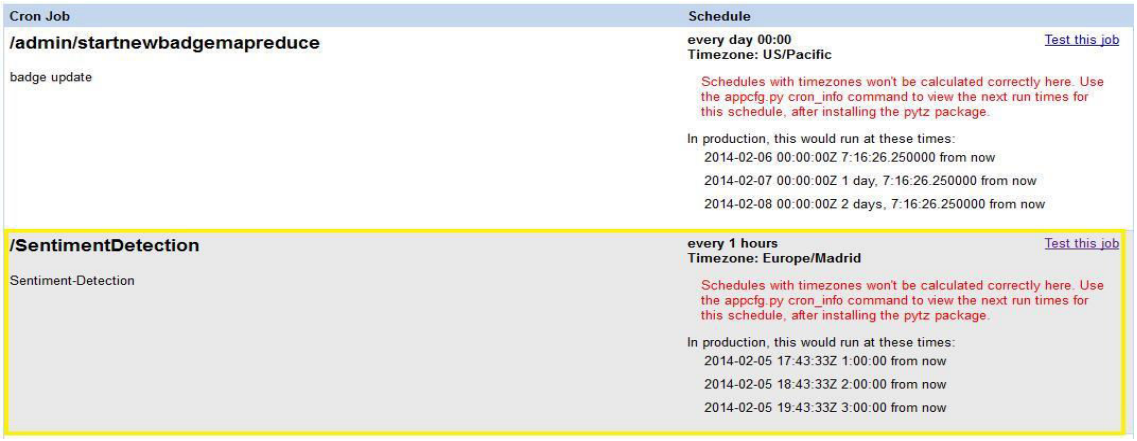

Figura 23. *cron job creado*

queue.yaml: Este es un fichero de configuración de una *task queue* o cola de tareas [24]. La Task Queue Python de App Engine pone a nuestra disposición un sistema para ejecutar trabajo en segundo plano. Se encarga de ejecutar las peticiones que no vienen del usuario, y lo hace de forma invisible para éste. La configuración que se ha usado en este trabajo se muestra a continuación:

```
queue:

- name: Sentiment-Detection
  rate: 15/s
```

El campo *name* especifica el nombre de la cola, que la aplicación utiliza cuando le añade una *task* o tarea. En *rate* se indica la velocidad media con la que se procesan las tareas en la cola. El valor de este campo es un número seguido de una barra y una unidad de tiempo (s para segundo, m para minutos y h para horas). En la figura 24 vemos como ha quedado configurada el sistema de encolado creado (recuadrado de amarillo). Si se pulsa el botón *Purge Queue* se eliminarán todas las tareas que queden pendientes en la cola.

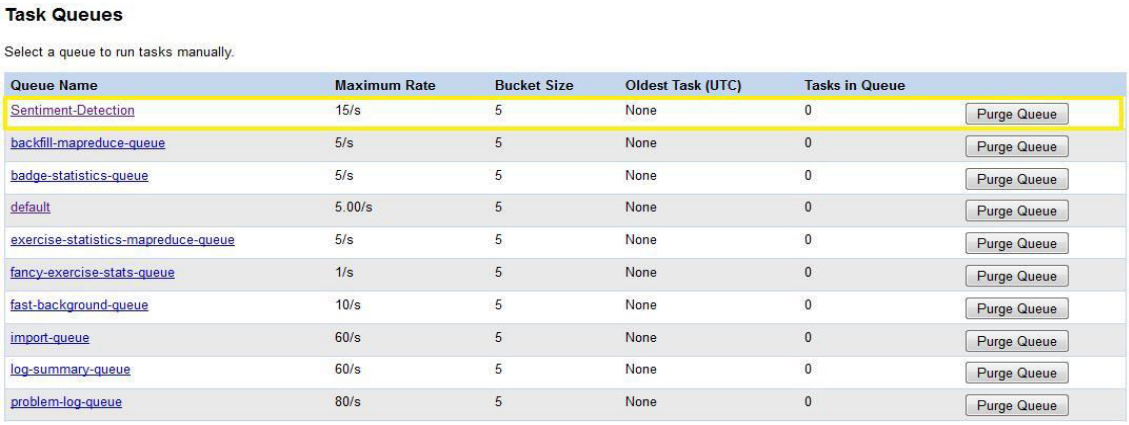

Figura 24. *Task Queue creada*

Unido al concepto de *cron job* y de *task queue* va el concepto de *task* o tarea. Una *task* es una pequeña porción de trabajo en App Engine, y al ejecutarse generan las medidas de los sentimientos de cada alumno.

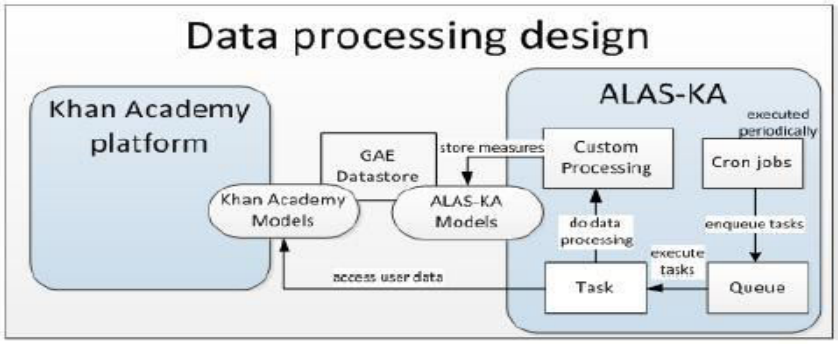

Figura 25. *Diagrama para el procesamiento de datos [1]*

En la figura 25 se puede ver el diseño para el procesamiento de datos seguido en ALAS-KA. Cuando se ejecuta el *cron job* genera una serie de *task.* Cada *task* generada se va añadiendo a la *queue* correspondiente, como en este caso solo se ha configurado una *queue*, todas las tareas van a la misma cola. Una vez añadidas a la cola, el sistema de encolado las irá ejecutando en el orden en que fueron añadidas. Si una tarea fallara, el sistema de encolado no se bloquearía, sino que enviaría a la tarea al final de la cola y seguiría ejecutando las demás tareas.

SentimentDetection.py: Se puede decir que este fichero python es la unión entre las clases de primer nivel y de segundo nivel. Como dijimos anteriormente, un *cron job* invoca una URL mediante una solicitud GET HTTP. Este fichero es el que recibe estas solicitudes GET HTTP.

```
class SentimentDetection(webapp2.RequestHandler):

    def get(self):   1

        studentUsers =  GqlQuery("SELECT * FROM ViewerUser WHERE student = True")     2

        logging.info("NOTE: Starting Sentiment Detection cron job");
        for studentUser in studentUsers:
            taskqueue.add(queue_name='Sentiment-Detection', url='/SentimentDetection', params={'userId': studentUser.user_id})   3

        logging.info("NOTE: Calculating mean value of Sentiment Detection");

        taskqueue.add(queue_name='Sentiment-Detection', url='/mean_sentiment', params={'studentUsers': studentUsers})   5

        logging.info("NOTE: Finished Sentiment Detection cron job");

    def post(self):   4
        user_id = self.request.get('userId')
        Frustration(user_id)
        Confusion(user_id)
        Boredom(user_id)
        Happiness(user_id)
```

Cuando *SentimentDetection.py* recibe la solicitud GET entra en su método *get* (**1**). Recupera todos los alumnos que haya (**2**), y obtenemos cada uno de ellos por separado mediante el bucle *for*. La función *taskqueue.add* (**3**) que está dentro del bucle se encarga de encolar las tareas en la cola llamada *Sentimen-Detection*. Para ello esta función envía una solicitud a una URL mediante el método POST HTTP, le pasa como parámetro la *Id* del usuario. Como se puede ver, la URL solicitada es la misma que solicita el *cron job*, pero la función en (**3**) utiliza el método POST, con lo que al llegar de nuevo a la clase *SentimentDetection.py*, se irá al método *post* (**4**). En (**4**) primero recuperamos el *ID* del usuario que pasamos como parámetro en (**3**), y después tenemos las cuatro funciones que calculan el nivel de sentimiento de un usuario.

Una vez se hayan encolado las tareas de todos los usuarios, se saldrá del bucle *for* y nos quedaría por encolar (**5**) la tarea que calcula la media de todos los sentimientos. Como vemos en el nombre de la cola, enviamos la tarea a la misma cola que antes, pero ahora el campo *url* cambia, con lo cual le dirigirá al método *post* de la clase *MeanSentiment.py* que veremos más adelante.

profileviewer_models.py: Como ya se mencionó en el apartado 2.5, la plataforma Khan Academy en su versión off-line ya traía creadas ciertas entidades. Esta clase crea tres entidades más que son necesarias para el desarrollo del trabajo, que son:

- ViewerUser: Esta entidad copia para cada usuario de la entidad *UserData* el *usuario,* el *Id* y el *nickname* del usuario, y además añade un campo propio que determina si un usuario es profesor o alumno.

- UserProfile: Esta entidad guarda para cada usuario de la plataforma los campos *usuario*, *Id* y *nickname* del usuario, *activity* (que indica si ha tenido actividad sobre la plataforma cada vez que se ejecuta un *cron job*), el nivel de los estados anímicos frustración, confusión, aburrimiento y alegría, y la hora exacta en la que se ha calculado cada uno de los estados anímicos.

- ClassMeanProfile: Esta entidad guarda el nivel medio de cada uno de los estados anímicos mencionados anteriormente de todos los alumnos.

main.py: Esta clase contiene todos los *handlers* o manejadores que traía desde un principio la versión off-line de la plataforma. A todos estos *handlers* hay que añadir otros cuatro que se han tenido que crear para el correcto funcionamiento del trabajo:

```
application = webapp2.WSGIApplication([

    ('/f', SentimentHandler.charts),                          1
    ('/SentimentDetection', SentimentDetection.SentimentDetection)  2
    ('/mean_sentiment', MeanSentiment.MeanSentiment),         3
    ('/combo', Combo.combo),                                  4
```

El *handler* **1** utiliza para mostrar las visualizaciones (esto se explicará más adelante). El *handler* número **2** lo utilizan el *cron job* y la función de encolado de *task* descritos anteriormente. El handler número **3** lo utiliza la función de encolado de *task* descrita anteriormente para encolar la tarea de calcular la media de los sentimientos. Y, por último, el handler **4** se utiliza para mostrar una página HTML con las visualizaciones de los sentimientos.

frustration.py, confusion.py, boredom.py y happiness.py: Estas clases calculan el nivel de sentimiento de frustración, confusión, aburrimiento y alegría respectivamente de cada alumno, siguiendo los modelos de detección explicados en el capítulo 3. Una vez calculados todos los sentimientos guarda el resultado y la hora en la que lo calculó en la entidad *UserProfile*.

MeanSentiment.py: Esta clase calcula la media de cada uno de los sentimientos y guarda el resultado en la entidad *ClassMeanProfile*. La media es calculada con respecto a los alumnos que hayan interactuado con la plataforma en el rango de tiempo sobre el que se calculan los sentimientos.

Combo.py: Esta clase es el inicio del conjunto de clases que permiten mostrar las visualizaciones. Cuando tenemos arrancada la plataforma off-line de la Khan Academy, y metemos en el navegador la url ***http://localhost:8080/combo***, esto crea una solicitud GET HTTP y el cuarto *handler* mostrado en *main.py* nos envía a la clase *Combo.py*.

```
class combo(webapp2.RequestHandler):

    def get(self):  1

        studentUsers =  GqlQuery("SELECT * FROM ViewerUser WHERE student = True")

        path = os.path.join(os.path.dirname(__file__), 'profile_viewer.html')  2

        template_values = {
                           'studentUsers': studentUsers,      3
                          }
        self.response.out.write(template.render(path, template_values))    4
```

Esta clase recibe la solicitud en su método *get* (**1**) y recupera todos los alumnos de la base de datos: En **2** se introduce la ruta del fichero HTML que va a mostrar la página donde poder elegir los usuarios en un desplegable. En **3** se introducen los parámetros que vaya a necesitar la plantilla HTLM para mostrar en el desplegable. Finalmente, en **4** se responde a la petición con la ruta y los parámetros necesarios, y esto nos lleva al fichero *profile_viewer.html*.

profile_viewer.html: Esta plantilla recibe los parámetros enviados por la clase *Combo.py* y muestra en una ComboBox (lista desplegable) todos los alumnos, para poder elegir de qué alumno quieres ver las visualizaciones con las estadísticas de los

sentimientos. En la figura 26 se muestra el desplegable con los alumnos creados en la versión off-line de la plataforma.

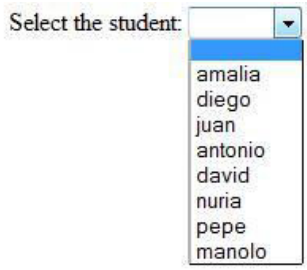

Figura 26. *ComboBox para seleccionar un alumno*

```
<script type="text/javascript">
    function setIframeSource() {

        var users_select = document.getElementById('userSelect'); 2
        selectedStudent = users_select.options[users_select.selectedIndex].value;
        var profileFrame = document.getElementById('profileFrame');
        var theUrl = '/f?userId=' + selectedStudent;
        profileFrame.src = theUrl;  3

    }
</script>
<body>
<div id="main_container">

   <center>
    <table border="0" width="95%" align="center" >
        <tr>

            <td>

                    <label for="users_select" >Select the student:</label>

                    <select name="userSelect" id="userSelect" onchange="setIframeSource()">
                        <option value="0"/>
                        {% for studentUser in studentUsers %}
                        <option value="{{ studentUser.user_id }}">          1
                            {{ studentUser.user_nickname }}
                        </option>
                        {% endfor %}
                     </select>
            </td>
```

En **1**, se crea la ComboBox, se recuperan los parámetros que nos han pasado y se introducen dentro del desplegable. En la zona sombreada (texto JavaScript), en el paso **2**, se recupera el alumno seleccionado en el desplegable, y en **3** se hace una solicitud al primer *handler* definido en *main.py* y se le pasa como parámetro el estudiante seleccionado.

SentimentHandler.py: La manera de actuar de esta clase similar a la de la clase *Como.py*. Contiene un método *get* que recibe la solicitud y recupera el alumno seleccionado en la ComboBox a través del parámetro que le han pasado. Después crea

la ruta a la plantilla HTML donde quiere enviar los parámetros (sentiment_line.html), guarda los parámetros a enviar, que en este caso van a ser los niveles de cada sentimiento y la hora en la cual se han calculado esos datos, y responde a la petición con la ruta y los parámetros guardados.

sentiment_line.html: Esta plantilla recibe los parámetros que le pasa la clase *SentimentHandler.py* y mediante el API de visualizaciones de Google Charts [21], muestra las estadísticas de los cuatro estados anímicos del alumno seleccionado en el desplegable. Para el desarrollo del código para mostrar las visualizaciones se ha vuelto a utilizar JavaScript dentro de la plantilla HTML.

4.3. Interfaces gráficas en la versión off-line de la plataforma

Se van a explicar las visualizaciones que se han creado y con las que se ha estado probando en la versión off-line de la plataforma Khan Academy antes de su integración en el módulo ALAS-KA.

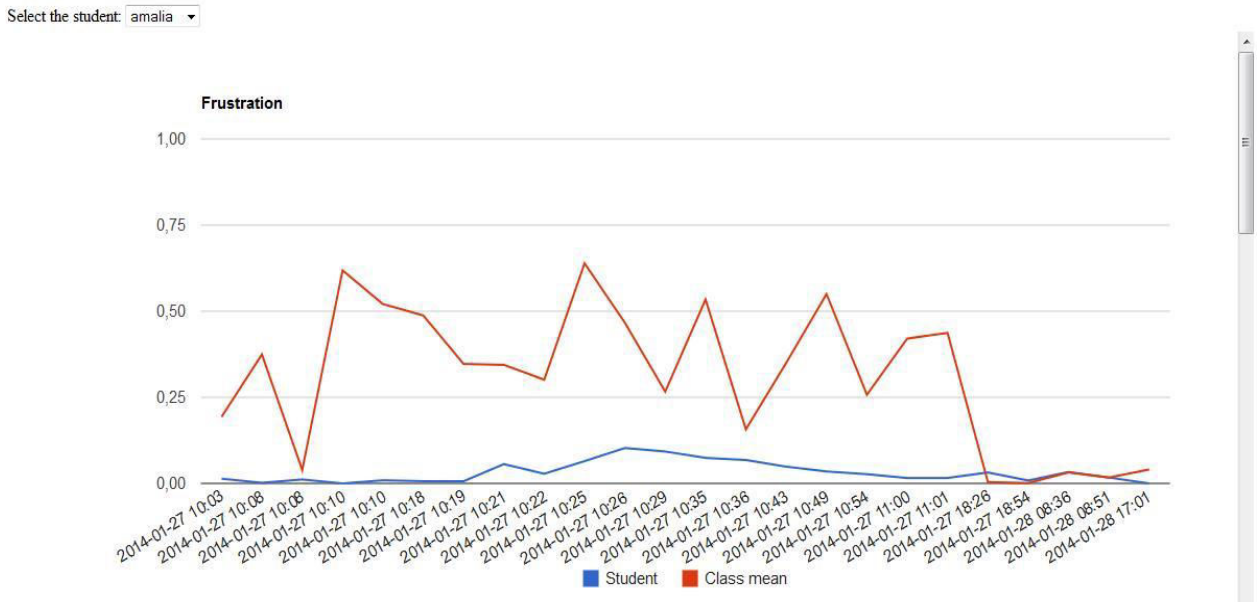

Figura 27. *Visualización de prueba en la versión off-line*

La figura 27 muestra la frustración del un estudiante. Como podemos ver, en el desplegable se ha seleccionado a *amalia* como alumno del que ver sus estadísticas. El nivel de frustración, que se muestra en el eje Y, puede variar entre 0 y 1 (al igual que el de todos los sentimientos). La línea azul muestra la frustración del alumno elegido, y la línea roja muestra la frustración de la media de la clase. Como se dijo en el apartado 4.2, los *cron jobs* se ejecutan cada hora, con lo que las horas mostradas en el eje X deberían ir todas igualmente espaciadas con un espacio de una hora entre ellas, pero

como dije, esto es una versión off-line en la que se han probados diversas cosas, y es por esto que las horas no están igualmente espaciadas. Por último, si desplazáramos hacia debajo el Scrollbar (barra de desplazamiento) de la derecha, podríamos ver las estadísticas de los demás sentimientos.

En la figura 28 se muestra otra visualización de prueba de otro estudiante y otro sentimiento. Como podemos observar el estudiante es *juan* y el sentimiento a representar es el de confusión. Si mantenemos el ratón sobre alguna de las medidas de cualquiera de las líneas (azul o roja), nos aparece la fecha en la que se calculó esta medida y el valor exacto del nivel de sentimiento en esa fecha.

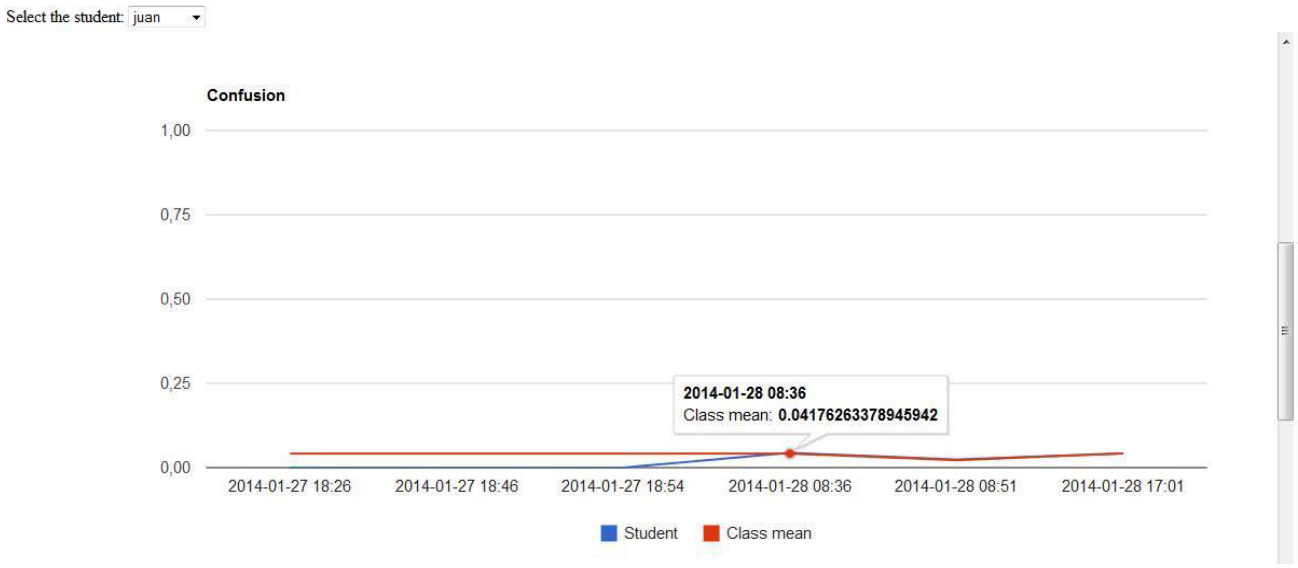

Figura 28. *Visualización 2 de prueba en la versión off-line*

Esta vez sólo se muestran seis medidas de la confusión pues, como se ha mencionado previamente, se estaban realizando pruebas de funcionamiento.

CAPÍTULO 5

INTEGRACIÓN, EVALUACIÓN Y VALIDACIÓN

En este capítulo se va a comentar el proceso de integración del módulo de detección de sentimientos en el módulo ALAS-KA, y se mostrarán visualizaciones de las pruebas de evaluación.

5.1. Integración

El proceso de integración del módulo de detección de sentimientos en el módulo ALAS-KA se hizo en un día. Para ello fue preciso contar con la ayuda de José Antonio Ruipérez, que diseñó e implementó el módulo ALAS-KA [7].

Se fueron introduciendo las clases una a una comprobando el correcto funcionamiento y cambiando pequeños detalles para que estuvieran dentro de la filosofía de ALAS-KA. En todos los *handlers* se añadieron derechos de administrador, pues en la versión off-line de la plataforma Khan Academy no era necesario ya que no se trabajaba sobre datos reales.

Por último, una vez introducidas todas las clases, se comprobó que todo estuviera igualmente relacionado que antes de la integración, pues se habían cambiado algunas cosas. Se comprobó que los *handlers* redirigieran a las clases indicadas, y una vez que todo estuvo bien integrado se pasó a las pruebas de evaluación y validación.

5.2. Evaluación y validación

Después del proceso de integración, se realizaron pruebas sobre datos reales. Estos datos provienen de las sesiones on-line que se realizan en el mes de Agosto en los cursos cero de la UC3M, donde un alumno puede encontrar vídeos y ejercicios que la universidad ha seleccionado a través de la Khan Academy. Debido a las normas de privacidad de la UC3M, no se ha tenido acceso directo a ningún usuario de los cursos cero, únicamente se ha podido aplicar el módulo de detección de sentimientos sobre el *user* (identificador de usuario que asigna la Khan Academy) de un alumno elegido aleatoriamente. En la figura 29, podemos apreciar el nivel de cada uno de los sentimientos de este alumno.

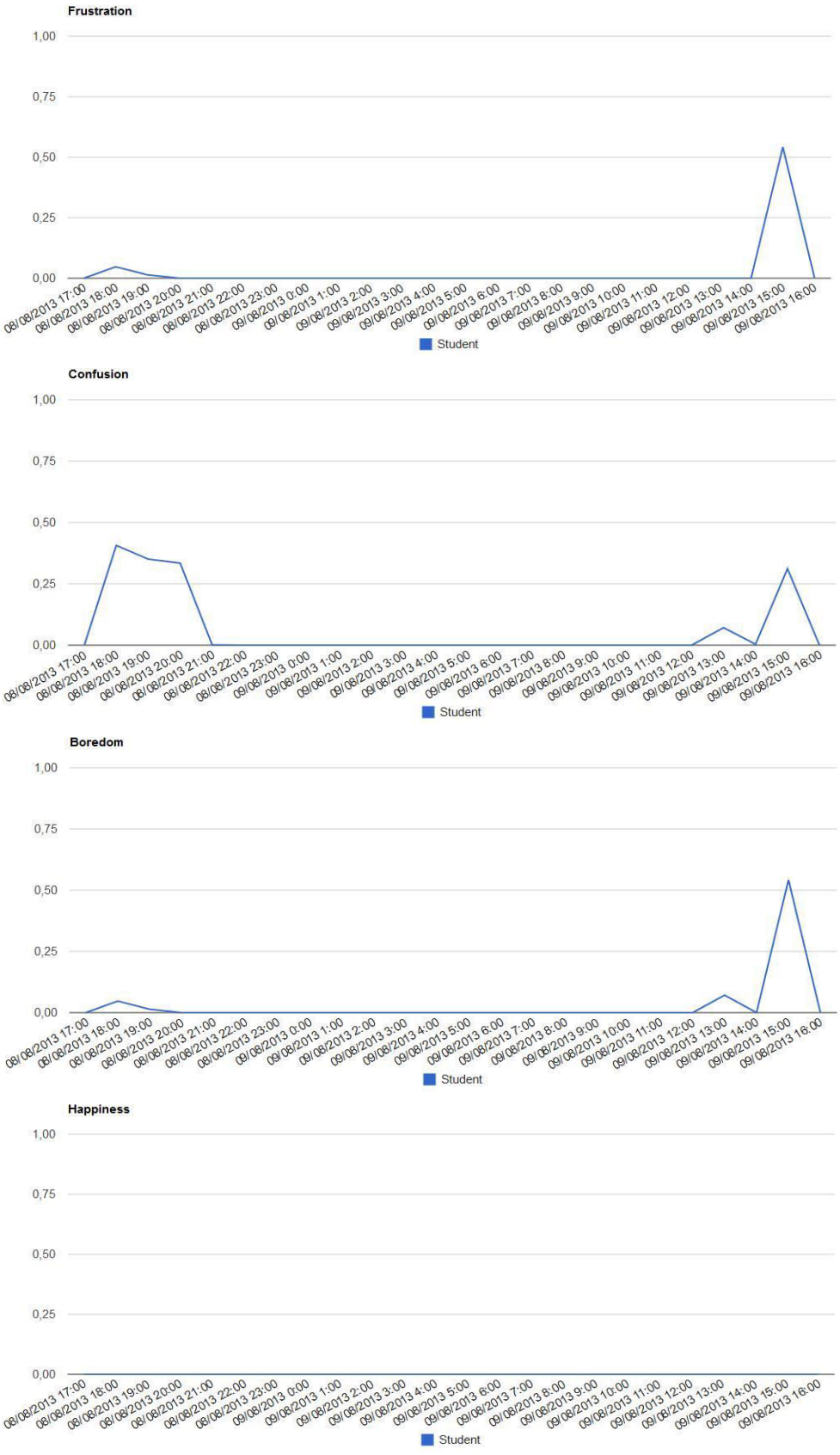

Figura 29. *Visualización de sentimientos sobre datos reales*

Como podemos observar en la visualización de *confusión*, el alumno ha estado activo entre las 17 y las 21 horas de día 8 de Agosto de 2013, y entre las 12 y 16 horas del día 9 de Agosto de 2013. El hecho de que haya estado activo en un margen de cuatro horas, no significa que haya estado 4 horas de seguidas haciendo ejercicios. Si por ejemplo, hacemos una medida de un sentimiento a las 21 horas, esta medida se calcula con los ejercicios realizados entre las 20 y las 21 horas, pudiendo haber estado durante el transcurso de esa hora sólo diez minutos activo.

Otro punto a comentar es el hecho de que a determinadas horas el nivel de sentimiento sea cero. Esto puede ser debido a dos causas, que en esas horas el alumno no haya hecho nada (lógicamente esto es lo que sucede en horario nocturno), o que habiendo estado activo, el cálculo del nivel de cierto sentimiento resulte cero. Este segundo caso es el sucedido en la visualización de *alegría*. Que el nivel de alegría sea cero para todas las horas, no quiere decir que el alumno haya resuelto mal todos los ejercicios sino que, como ya se mencionó para este sentimiento, el alumno no ha conseguido obtener ningún *badge*.

CAPÍTULO 6

CONCLUSIONES Y TRABAJO FUTURO

En este capítulo se hablará de las conclusiones extraídas y de los posibles trabajos futuros.

6.1. Conclusiones

Este proyecto nace con el propósito de hacer más fácil a un profesor la tarea de comprender cómo se sienten ciertos alumnos respecto a ciertos ejercicios. La gran cantidad de alumnos que un profesor puede tener a cargo, hace que el centrarse en cada uno de ellos de manera individual se antoje imposible. Sin embargo, mediante las visualizaciones de sentimientos, un profesor puede, de una manera relativamente rápida, detectar qué ejercicio ha resultado más difícil respecto a una clase o a un alumno en concreto.

Se considera que los objetivos propuestos al comienzo del proyecto se han conseguido de manera satisfactoria. Se ha implementado un módulo donde se detectan cuatro sentimientos, los cuales se muestran mediante visualizaciones medidos de cero a uno. Estas visualizaciones muestran los sentimientos de cada alumno por separado junto con la media de la clase. Todo esto se ha desarrollado obteniendo datos de una versión off-line de la plataforma Khan Academy, y posteriormente mediante la integración del módulo de detección de sentimientos en el módulo ALAS-KA.

Para el logro de estos objetivos se ha tenido que aprender y hacer uso de diversas tecnologías como el lenguaje de programación Python, HTML, JavaScript, Google App Engine, bases de datos de Google, Google Charts, etc.

6.2. Trabajo futuro

Existe un conjunto de tareas que sería posible realizar en un futuro con el objetivo de continuar con lo expuesto en este trabajo:

- Tener en cuenta nuevos parámetros para realizar las medidas, como los vídeos vistos por cada alumno, si en un ejercicio ha utilizado las pistas, si se ha personalizado su perfil, etc. Como ya se mencionó, en este trabajo solo se han utilizado los ejercicios y los badges para conseguir las medidas.

- Calcular medidas de más sentimientos aparte de los que aparecen en este trabajo, para poder relacionarlos y sacar más conclusiones.
- Realizar un análisis exhaustivo de las medidas obtenidas, e intentar razonar el por qué de cada medida, o por qué en unas asignaturas predomina un sentimiento y en otras otro diferente, etc.
- Modificar los diagramas de cálculo de emociones expuestos para poder aplicarse en otras plataformas.

BIBLIOGRAFIA

[1] J. A. Ruipérez-Valient, P. J. Muñoz-Merino and C. Delgado Kloos. *"An architecture for extending the learning analytics support in the Khan Academy framework"*

[2] P. J. Muñoz Merino, J. A. Ruipérez-Valient and C. Delgado Kloos. "*Inferring higher level learning information from low level data for the Khan Academy platform"*

[3] Plataforma Khan Academy - *https://www.khanacademy.org/*

[4] J. Manyika, "Big Data: The Next Frontier for Innovation, Competition, and Productivity" Executive Summary, McKinsey Global Institute, May 2011

[5] J.P. Campbell and D.G. Oblinger. 2007. *"Academic Analytics. EDUCAUSE Quarterly"*. October (2007).

[6] D. Leony, P. J. Muñoz-Merino, A. Pardo, C. Delgado Kloos. *"Provision of awareness of learners' emotions through visualizations in a computer interaction-based environment" 2013*

[7] D. Clow. *"The learning analytics cycle: closing the loop effectively"*. *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*. Pages 134-138.

[8] J. A. Ruipérez Valiente – *"Diseño en implementación de un módulo de analítica de aprendizaje en la plataforma Khan Academy".* Trabajo Fin de Master 2013

[9] R. W. Picard. *"Affective computing"*. MIT press, 2000.

[10] D. Leony, P. J. Muñoz-Merino, A. Pardo, C. Delgado Kloos. *"Modelo basado en HMM para la detección de emociones a partir de interacciones durante el aprendizaje de desarrollo de software" 2013*

[11] I. Arroyo, D. G. Cooper, W. Burleson, B. P. Woolf, K. Muldner, and R. Christopherson, *"Emotion sensors go to school"*, in Proceeding of the 2009 conference on Artificial Intelligence in Education, July 6th-10th, Brighton, UK, IOS Press, 2009, pp. 17–24.

[12] S. K. D'Mello, S. D. Craig, J. Sullins, and A. C. Graesser, *"Predicting affective states expressed through an emote-aloud procedure from autotutor's mixed-initiative dialogue"*, International Journal of Artificial Intelligence in Education, vol. 16, no. 1, pp. 3–28, 2006.

[13] S. D'Mello, A. Graesser, and R. W. Picard, *"Toward an affect-sensitive autotutor"*, Intelligent Systems, IEEE, vol. 22, no. 4, pp. 53–61, 2007.

[14] A. Pardo and C. Delgado Kloos, *"Stepping out of the box. Towards analytics outside the Learning Management System"*, in International Conference on Learning Analytics and Knowledge, 2011.

[15] V. A. Romero Zald´ıvar, A. Pardo, D. Burgos, and C. Delgado Kloos, *"Monitoring Student Progress Using Virtual Appliances : A Case Study"*, Computers & Education, vol. 58, no. 4, pp. 1058–1067, 2012.

[16] S. K. D'Mello, S. D. Craig, J. Sullins, and A. C. Graesser, *"Predicting affective states expressed through an emote-aloud procedure from autotutor's mixed-initiative dialogue"*, International Journal of Artificial Intelligence in Education, vol. 16, no. 1, pp. 3–28, 2006.

[17] Python versión 2.7 - *http://www.python.org/download/releases/*

[18] Documentación en Wikipedia - *http://es.wikipedia.org/wiki/HTML*

[19] Documentación en Wikipedia - *http://es.wikipedia.org/wiki/JavaScript*

[20] Documentación sobre Google App Engine
https://developers.google.com/appengine/docs/whatisgoogleappengine?hl=es

[21] Documentación sobre App Engine Datastore
https://developers.google.com/appengine/docs/python/datastore/?hl=es

[22] Documentación sobre Google Charts
https://developers.google.com/chart/interactive/docs/index?hl=es

[23] Documentación sobre tareas programadas en Google App Engine
https://developers.google.com/appengine/docs/python/config/cron?hl=es

[24] Documentación sobre colas de tareas en Google App Engine
https://developers.google.com/appengine/docs/python/config/queue?hl=es